DUDLEY KNOX LIBRARY
NAVAL POSTGRADUATE SCHOOL
MONTEREY, CA 93943-5101

NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

SOFTWARE AGENTS AND THE DEFENSE INFORMATION INFRASTRUCTURE: REENGINEERING THE ACQUISITION PROCESS

by

Jerome Hudson

September 1998

Thesis Advisor:	Mark E. Nissen
Co-Advisor:	Tung X. Bui

Approved for public release; distribution is unlimited.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	September 1998	Master's Thesis

4. TITLE AND SUBTITLE
SOFTWARE AGENTS AND THE DEFENSE INFORMATION INFRASTRUCTURE: REENGINEERING THE ACQUISITION PROCESS

5. FUNDING NUMBERS

6. AUTHOR(S)
Hudson, Jerome

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Naval Postgraduate School
Monterey, CA 93943-5000

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

10. SPONSORING / MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES
The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

12a. DISTRIBUTION / AVAILABILITY STATEMENT
Approved for public release; distribution is unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT *(maximum 200 words)*

Process innovation within the Department of Defense (DoD) procurement system ultimately translates into flexibility, combat effectiveness and technological advantage on the modern battlefield. A critical enabler of process innovation is the effective use of advanced information technology (IT) products, such as software agents. Software agent-based systems are used as an IT enabler for redesigning processes within the Defense Information Infrastructure (DII) Acquisition system. The Simplified Acquisition Procedures (SAP), a key element of acquisition reform, are used as the focus of our redesign efforts. To accomplish this task, the process is represented using a traditional process-flow model, Use Case analysis to integrate the DII macro-process view and the agent technology micro-view, and using a heuristic measure of process complexity to identify processes suitable for machine verses human performance. By exploiting the inherent strengths of both software and human agents, productivity is enhanced by freeing human agents from routine tasks and enables the refocusing of human resources to high value acquisitions. The result is an agent-based redesign of SAP processes where human agents and software agents share in the responsibilities for process execution.

14. SUBJECT TERMS
Software Agents, Acquisition Reform, Process Innovation, Defense Information Infrastructure

15. NUMBER OF PAGES
126

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFI- CATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	UL

SOFTWARE AGENTS AND THE DEFENSE INFORMATION INFRASTRUCTURE: REENGINEERING THE ACQUISITION PROCESS

Jerome Hudson
Major, United States Army
B.S., Prairie View A&M University, 1984

Submitted in partial fulfillment
of the requirements for the degree of

MASTER OF SCIENCE IN MANAGEMENT

from the

NAVAL POSTGRADUATE SCHOOL
September 1998

ABSTRACT

Process innovation within the Department of Defense (DoD) procurement system ultimately translates into flexibility, combat effectiveness and technological advantage on the modern battlefield. A critical enabler of process innovation is the effective use of advanced information technology (IT) products, such as software agents. Software agent-based systems are used as an IT enabler for redesigning processes within the Defense Information Infrastructure (DII) Acquisition system. The Simplified Acquisition Procedures (SAP), a key element of acquisition reform, are used as the focus of our redesign efforts. To accomplish this task, the process is represented using a traditional process-flow model, Use Case analysis to integrate the DII macro-process view and the agent technology micro-view, and using a heuristic measure of process complexity to identify processes suitable for machine verses human performance. By exploiting the inherent strengths of both software and human agents, productivity is enhanced by freeing human agents from routine tasks and enables the refocusing of human resources to high value acquisitions. The result is an agent-based redesign of SAP processes where human agents and software agents share in the responsibilities for process execution.

TABLE OF CONTENTS

I. INTRODUCTION

> We will need organizations and processes that are agile enough to exploit emerging technologies and respond to diverse threats and enemy capabilities.
>
> Joint Vision 2010 - Agile Organizations [Ref. 15]

> We will need a responsive research, development and acquisition process to incorporate new technologies. This process must leverage technology and management innovations originating in the private sector through responsive access to commercial developments
>
> Joint Vision 2010 - Enhanced Material [Ref. 15]

Process innovation within the Department of Defense (DoD) procurement system ultimately translates into flexibility, combat effectiveness and technological advantage on the modern battlefield. Process innovation is defined as a fundamental rethinking of business processes and the application of change methodologies in order to create orders-of-magnitude increases in productivity [Ref. 2]. A critical enabler of process innovation is the effective use of information technology. Therefore, it is imperative that advanced information technology products, such as software agents, be exploited and systematically introduced as an integral part of any strategy to transform and innovate our procurement system. Process innovation and acquisition reform enabled by open, robust, adaptable and intelligent information technologies are needed in order to provide

technologically superior, cost effective and timely solutions to our operational warfighting elements.

A. BACKGROUND

In his thesis work, Mark E. St. Moritz cites S.N. Sherman's book, *Government Procurement Management: Special Edition* and writes:

> Sherman states that, '[t]he computerization of government procurement programs have evolved slowly.' He goes on to say that most advances in the automation of the acquisition process are the result of individual effort and not the result of any significant agency initiatives. From a policy point of view, more effort is devoted to 'procurement controls, ethics, policy, and audit than automation.' The adoption of information technology by the government does not match the progress achieved by private industry. [Ref. 32: pages 27-28]

Even though the vast majority of acquisition reform efforts are centered on non-automation specific programs and activities, there are a significant number of automation related efforts moving forward. In any case, the real issue presented by Sherman is not the lack of automation resources within the acquisition system, but the lack of a clearly defined agency-wide enterprise solution. There is strong evidence that this is about to change.

A number of statutory and DoD initiatives suggest that information technology initiatives, based on achieving process innovation within the DoD acquisition system, are being taken very seriously by our Acquisition leadership. As indicated below, the comments taken from a fact sheet produced by the newly formed Office for Logistics/Life

Cycle Information Integration (L/LCIIO) indicate a major paradigm shift in the management of information technology within the acquisition business function:

> The recent implementation of the Information Technology Management Reform Act within the Department of Defense (DoD), the President's July 1, 1997 executive memorandum on electronic commerce, and DoD Acquisition & Technology's (A&T's) initiatives to go paperless in many of DoD's functional and cross-functional processes, have made it a necessity to focus on integrating cross-functional information management strategies, functions, plans and resources within A&T. ...
>
> ... Mr. Longuemare designated Michael J. Mestrovich as the A&T Information Management Executive (IME) responsible for life cycle information management program oversight. Mark Adams was named to lead the recently created Life Cycle Information Integration Office (LCIIO). ... Mestrovich and Adams will co-chair an Overarching Integrated Process Team (OIPT) that will take a cross-functional approach that better utilizes existing systems to provide faster and greatly improved customer service. [Ref. 21]

Information technology, as an enabling technology for process innovation, has also assumed a leading role. The designation of Dr. Mestrovich as the DUSD (A&T) Information Management Executive (IME) and the creation of the LCIIO are clear evidence that DoD acquisition is moving toward a comprehensive enterprise solution.

The DUSD (A&T)'s goal of "integrating cross-functional information management strategies, functions, plans and resources" is certainly based on an effort to align the acquisition business function with the Defense Information Infrastructure (DII). This integration effort, within the acquisition business function, seeks to provide to the warfighter [Ref. 26]:

- Increased Responsiveness
- Improved Asset Visibility
- Agile Infrastructure
- Reduced Inventories
- Faster Acquisition Cycle Times
- Greater Competition
- Lower Costs

A number of mission applications and initiatives are being developed and deployed to meet the stated goals and bring about acquisition reform through information technology. Some of the efforts that have the most far-reaching impact are listed below [Ref. 21]:

- **Federal Acquisition Computer Network (FACNET)** - The FACNET is an (Electronic Data Interchange) EDI-based architecture for electronic commerce mandated by the Federal Acquisition Streamlining Act (FASA) of 1994.
- **Joint Computer-Aided Acquisition and Logistics Support software (JCALS)** - JCALS is a system that provides an infrastructure capable of integrating digitized technical data, which supports weapons systems acquisition and the logistics life cycle. The system is data-driven and provides an automated information system independent of application.
- **Standard Procurement System (SPS)** - SPS is a standardized automated procurement system being developed for use by the DoD procurement community. It is the next generation of procurement application software designed to link acquisition reform and common DoD procurement business processes with commercial best practices and advances in electronic commerce.

- **Centralized Contractor Registration (CCR)** - The CCR is the primary DoD repository for contractor information required for the conduct of business with the DoD. The database consists of information pertinent to procurement and financial business transactions.
- **Past Performance Automated Information System (PPAIS)** - The PPAIS is an automated system used to collect and provide access to information about contractors' past performance which may be used by DoD acquisition personnel.
- **Paperless Acquisition Initiative** - An internal DoD business process improvement initiative, launched by the former Defense Department Comptroller, John J. Hamre, calls for a "totally paper-free contract writing, administration, finance, and auditing process" by January 1, 2000. It is an umbrella initiative that will incorporate ongoing initiatives for the use of purchase cards, electronic catalogs, electronic commerce, and imaging.
- **E-Mall** - The DoD E-Mall is designed to provide a seamless user interface with existing and future enabling electronic commerce technologies. The initial architecture will involve the design and establishment of an Electronic Commerce Infrastructure (ECI), which would allow contractors to send and receive purchasing information electronically to and from Government procurement offices.

- **Electronic Smart Card** - The smart card is credit card sized memory storage or a microprocessor controlled device that provides a convenient means of storing, securing and processing information. For example, Smart cards can act as access control devices by holding and transferring private electronic keys. Or, they can act as devices for the purchase of goods or services based on stored values and interacting with networked financial institutions.

These systems and initiatives are aligned with the purpose of providing DoD with enterprise level cross-functional electronic commerce capabilities that will lead information technology based acquisition reform well into the 21st Century.

The realignment and consolidation of management functions and the multi-function focus within the DUSD (A&T) are creating a consistent enterprise computing vision within the DII acquisition business function. The integration of mission applications within a common data-sharing environment is seeking to create an infrastructure that allows the acquisition business function to be more agile, economical and efficient. But, the DoD needs to go further. The IT-based advances discussed above only take DoD to the level of current technology and practice. As such they establish the necessary IT infrastructure for process innovation, but they seem to ignore advanced and emerging technologies offering tremendous potential to effect the kinds of orders-of-magnitude performance improvements sought for DoD acquisition. One particularly promising and exciting technology involves the use of software agents to perform autonomous work tasks on behalf of their owners. This emerging and rapidly advancing

area holds great promise for innovating Defense acquisition. Yet few have studied this technology for application to the procurement process.

B. PURPOSE

The purpose of this thesis is to provide insights and recommendations for the use of software agent technologies within the DII, with a specific focus on the acquisition business function. The thesis seeks to determine how agent technologies can be embedded within the acquisition domain and what benefits, in terms of process improvement, can be expected through their application. The exploitation of software agents, as an advanced information technology appears to offer excellent opportunities for process innovation and Defense acquisition represents a domain that is ripe for their application.

C. RESEARCH QUESTIONS

The primary research question addressed through this research is:

How can current software agent technologies enhance and streamline the DoD acquisition process?

The secondary questions are as follows:

- What are Software Agents?
- How do agents interact to perform tasks?
- What is the current state of agent technology?
- How can agents support organizational processes?

- What benefits can be achieved/expected through agent-based acquisition?
- How can a specific acquisition process be redesigned through agent technologies?

D. SCOPE AND LIMITATIONS

This study of software agents in the context of DoD procurement shows that software agents can act as collaborative knowledge-based autonomous "workers." As "workers," software agents provide a foundation for intelligent systems that provide information filtering, workflow management and task automation.

Software agents are in their infancy. Architectures, coordination mechanisms, agent-based software engineering methodologies and a simple universally agreed upon definition are all in a state of flux and vigorous debate. Therefore, we limit our research concerning agent capabilities to agent technologies that are currently available within the commercial sector and those that have a proven implementation within academia. The intent is for this research to be quite applied in nature, as theoretical research is only used to provide clarity and to document the direction of current efforts.

E. APPROACH

An extensive literature search of recent books, journals, periodicals and World-Wide-Web (WWW) resources in the following related disciplines and areas of study provide the foundation for this research:

- DoD Procurement
- DoD Policies
- Technical Standards
- Management Systems
- Management Information Systems
- Object Oriented Design
- Distributed Artificial Intelligence
- Software Agents and Agent Systems

- Knowledge and Expert Systems
- Knowledge Representation

The literature search provides a basis of analysis in determining how software agents, through the DII, can enhance and streamline the acquisition process. Analyzing both the process benefits and the technological issues relating to incorporating agent technologies requires an iterative approach.

An iterative analysis process is used that oscillates between a macro-process view and a micro-technology view. The macro-process view involves defining and analyzing the system domain features and functions, candidate agent roles, tasks, processes, objects, domain knowledge and the relationships between entities in order to determine benefits of applying agent technologies to the acquisition system. The micro-technology view focuses on technical issues related to specifying implicit behavioral attributes necessary for candidate agents within the acquisition system.

F. ORGANIZATION OF THE STUDY

Chapter I provides background on the doctrine, policy and reform initiatives currently shaping the DoD Acquisition enterprise. It also specifies the purpose, scope, research questions, approach and organization of this thesis. Chapter II provides an introduction to the DII and an overview of the objective implementation. It covers the enterprise concepts and technological principles that provide the context and environment for incorporation of software agents. This chapter concludes with an examination of the impact that the DII will potentially have on acquisition reform efforts and its relevance to incorporating agent technologies. Chapter III provides an overview of agent concepts and

technologies. It provides a basic treatment of some key issues relating to agent theories, architectures and systems. The chapter also includes a brief examination of the current state of agent technologies. It provides the necessary technical foundation in software agent concepts and technologies that will be used to support the analysis of incorporating software agents within the DII. Chapter IV specifically addresses the use of agent-based systems to redesign acquisition processes. Chapter V provides a brief summary of the thesis and makes recommendations for future research.

II. THE DEPARTMENT OF DEFENSE INFORMATION INFRASTRUCTURE

A. INTRODUCTION

The DII represents a global vision of enterprise computing within the DoD. This vision seeks to define a global environment of communication networks, computers and interoperable data sharing information systems, and provides seamless integration of DoD business functions. Its technological approach and principles will guide every information technology effort within DoD, well into the foreseeable future.

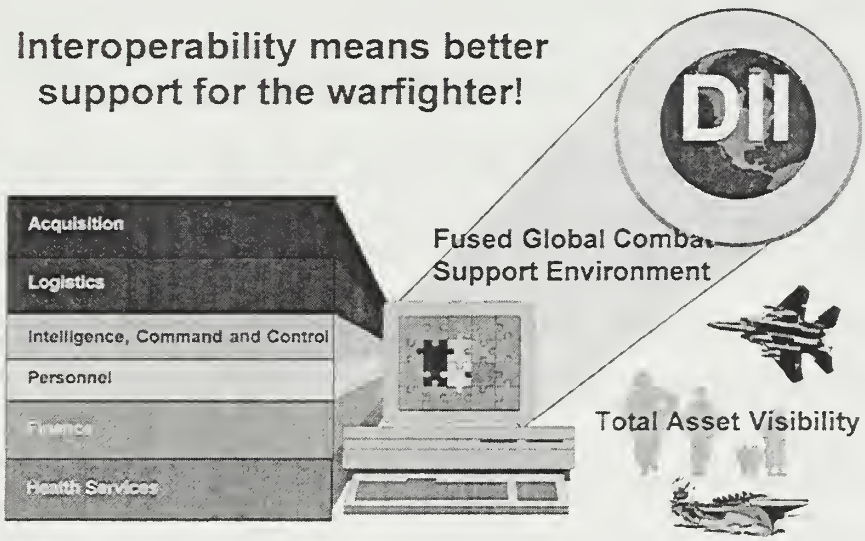


Figure 1: The DII Vision

It is a concept whose evolution is rooted in Joint Vision 2010 and the DoD Command, Control, Communications, Computers and Intelligence for the Warrior

(C4IFTW) concept. The refinement, integration and evolution of these two concepts have led to the global enterprise vision of a web of interconnected and interoperable information systems across all DoD business functions, including acquisition.

In this chapter, we define the DII as that vision of DoD enterprise computing. In discussing the DII, we provide the macro-view of the DOD computing environment in which agent technologies will operate in the future. As a primary focus, we look at the implications of the DII, as a DoD enterprise strategy for acquisition reform and in integrating software agent technologies.

B. BACKGROUND

In 1996, the Chairman of the Joint Chiefs of Staff (CJCS) released the Joint Vision 2010, a conceptual document that provided the vision for joint warfighting in the near future [Ref. 15]. The vision was exceptional in that it provided a conceptual template centered on new operational concepts (e.g., Dominant Maneuver, Precision Engagement, Focused Logistics, Full Dimensional Protection) and a framework in which the individual services (e.g., Army, Navy, Air Force) would define doctrine and acquisitions over the subsequent 15 years.

A cornerstone in the Joint Vision 2010 vision is the use of advanced information technology products in attaining information superiority. Information superiority is "the capability to collect, process, and disseminate an uninterrupted flow of information while exploiting or denying an adversary's ability to do the same" [Ref 15]. The use of

information technologies to provide decision makers with accurate, timely and relevant information is critical to the rapid execution of joint operations in the future.

But what does the implementation of an information infrastructure that allows for the execution of the Joint Vision 2010 look like? In fact, the conceptual model that articulates the information infrastructure requirements for future joint operations was documented prior to the release of Joint Vision 2010. The C4IFTW concept is formally documented for the services in Joint Publication 6-0 in 1995. It is based on the C4I for the Warrior conceptual document released in 1992 by the C4I Architecture and Integration Division (J6I) of the Joint Staff [Ref 16].

The C4IFTW concept promotes a vision of creating a fully integrated system of interoperable functional applications. Its ultimate goal is to provide the warfighter with a synergistic capability that provides "[a] fused real time, true representation of the Warrior's battlespace - an ability to order, respond and coordinate horizontally and vertically to the degree necessary to prosecute his or her mission in that battlespace." [Ref. 16: page 4]

The C4IFTW concept proposes a multi-tiered distributed architecture that is the antithesis of the proprietary mainframe based Worldwide Military Command and Control system (WWMCCS) of the day. In part, the realization of the C4IFTW vision would occur in the objective system design of the Global Command and Control System (GCCS), the WWMCCS replacement.

The GCCS program, modeled after the C4IFTW concept, is focused on achieving an information infrastructure that is capable of integrating various functions and

providing a means to share and integrate data across the enterprise. It represents an adaptable and flexible architecture that is seeking to incorporate open standards, a client-server computing model, and a common operating environment that provides mechanisms for incorporating new functionality.

The power of the GCCS systems development approach is recognized as being applicable beyond the C4I community. It is noted that the underlying strategies and concepts can provide an excellent capability for all DoD mission functions [Ref. 6]. Specifically, the expansion of the GCCS Common Operating Environment (COE) into the DII COE is the first step in creating a powerful vision of global computing within DoD.

C. A DOD ENTERPRISE STRATEGY

The DII can best be described as an enterprise strategy. This strategy ultimately seeks, as its principal goals, to provide a common operational picture (COP) and a common management picture (CMP) to the warfighter and to all DoD decision-makers worldwide. Although its central focus and goals are still rooted in supporting the warfighter, the DII extends well beyond the operational, C4I and mission applications environment. It articulates a global enterprise vision and strategy for all of DoD.

1. The DII Enterprise Vision

The formal documentation of policies, technologies and structure of the DII are still evolving, but its scope clearly presents a comprehensive framework for DoD

enterprise computing, a framework based on sound design principles and open technical standards. As noted in the DII Master Plan, the

> Defense Management Report Decision (DMRD) 918C created the DII not as a single program, but as a capability resulting from the integration of individual information management programs across the DoD to:
>
> (1) revolutionize information exchange Defense-wide,
>
> (2) strengthen our ability to apply computing, communications, and information management capabilities effectively to the accomplishment of DoD's mission,
>
> (3) significantly reduce the information technology burdens on operational and functional staffs, and
>
> (4) enable the operational and functional staffs to access, share, and exchange information world-wide with minimal knowledge of communication and computing technologies.
>
> Simply put, the DII is to provide seamless, secure information products and services to DoD users, especially warfighters, in support of decision-making and mission accomplishment. [Ref. 5]

The DII is an integration effort aimed at creating a ubiquitous information services infrastructure stretching from the foxhole to the highest levels of the DoD decision-making hierarchy. It is a global enterprise-computing environment and a vision of distributed computing that will effect every aspect of the DoD, including its acquisition system.

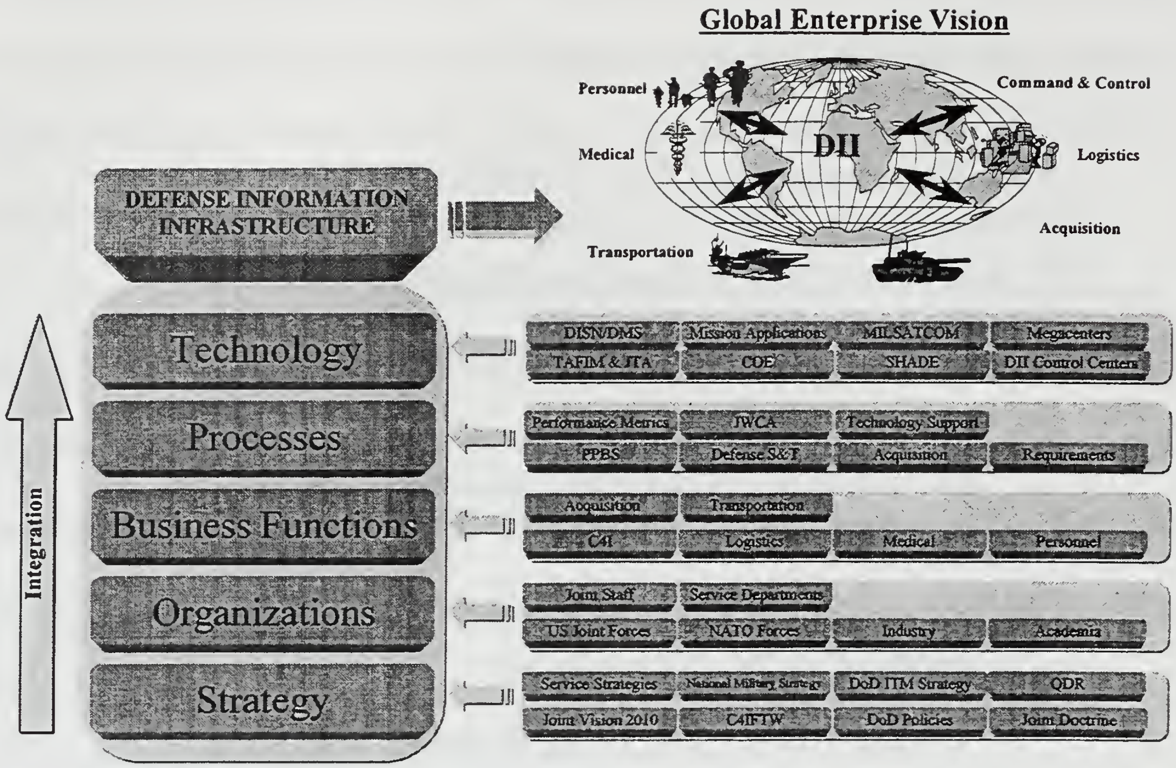


Figure 2: The DII Enterprise Integration Effort

The DII enterprise represents the technical infrastructure, management processes, organizations, strategies and business functions. It is the integration of all these elements that make the DII truly a DoD enterprise solution. Each element is critical in achieving the overall objectives of the DII vision. But, it's the technical strategy that provides the greatest value to our immediate concerns and research.

2. DII Technical Strategy

The technical integration effort, defining the scope of the DII, includes

> …the web of communications networks, computers, software, databases, applications, weapon system interfaces, data, security services, and other services that meet the information processing and transport needs of DOD users, across the range of military operations. [Ref. 5: page 2-1]

In order to integrate these various components, a technical strategy is defined based primarily on the Technical Architecture Framework for Information Management (TAFIM), which is the DOD Technical Reference Manual (TRM) for information management systems and the Joint Technical Architecture (JTA) which specifies DII compliance. Although the DII technical strategy, in terms of specific technologies, is still evolving, the underlying technical design principles and concepts used in the specification of those technologies are fixed. The guiding design principles and concepts are derived directly from the operational requirements of Joint Vision 2010 and the vision of the C4IFTW concept paper.

Based on the shortcomings inherent in the existing C4I systems and the operational requirements of Joint Vision 2010, C4IFTW specifies some innovative information system concepts critical to achieving the objective vision. The DII technological design is founded on an "open" systems approach, a client-server architecture, a common operating environment (COE), a shared data environment (SHADE) and evolutionary capability enhancements.

Open systems are developed in accordance with published standards and specifications that are open to the public and generally maintained by a governing standards body. This approach to systems design is beneficial for a number of reasons. First, open systems provide a means for vendors to provide products that are compatible and interoperable based on standard protocols and interfaces. Second, and probably the most important benefit, DoD will have a wide variety of choices in selecting software and

hardware components. Third, some systems can literally be "plugged in" and instantly provide the desired functionality. Finally, openness benefits DoD by encouraging competition and promoting technical innovation.

Client-server computing is based on the notion of using network technologies to distribute processing, data and application services down to the organization or functional units. It allows greater control and responsiveness to the dynamic conditions of the business environment and promotes agility and flexibility in system maintenance processes. It also provides greater fault tolerance than mainframe computing since there is built-in system redundancy with no single point of failure. Mainframe computing is the antithesis of Client-Server computing. It places processing, presentation, data services and application services at a central location.

The pace of change within the business and Joint Service environments requires units to rapidly adapt to changing conditions. Supporting their efforts, they require a responsive and flexible information services infrastructure and organization. The centralized structure of the mainframe computing environment forces a sharing of resources and prohibits a flexible response to the needs of any single organizational element. This in turn creates tension between the central organization and the business units.

Client-server computing offers a powerful solution. It allows local processing of applications and presentation services, through client machines, with a centralized server machine providing network management, database access and centralized file services. This approach accommodates the centralized control of the infrastructure and data

standards, while enabling dispersed organizations to control the business rules and logic of the applications environment.

The DII client-server architecture is closely aligned with the original C4IFTW concept. The C4IFTW architecture is broken into three hierarchical categories: the Warrior Terminal, the Warrior Battlespace and the Infosphere[1]. But, where the client-server model defines a technical architecture, the C4IFTW categories represent a logical architecture of functional, informational, service and system requirements based on the needs of the warfighter.

The Warrior terminal, aligned with the presentation layer of the client server model, is envisioned as a platform allowing a warrior access to fused multi-media information, anytime from anywhere in the world with the same human-computer interface (HCI). Each terminal provides the warrior with real time decision support aids that incorporate "artificial intelligence and decision support systems, rapid information fusion from intelligence sources and forces, virtual reality, integrated and flexible rapid planning tools, wargaming and simulation, and multimedia technologies" [Ref. 16:, page 9].

The second tier, much like the business function layer of the client server model, is defined as the "Warrior's Battlespace." It defines in essence the informational and system requirements that are necessary for the warrior to clearly view and coordinate within the battlespace. This provides the warrior with an integrated picture of the ground,

---

[1] Infosphere - A term used within the C4IFTW concept paper that refers to the worldwide network of military and commercial communications systems, networks, and data warehouses.

air, sea, space and special operations activities within the area of interest [Ref. 16:, page 9].

This intermediate layer supports reusable components that are accessible from the Warrior terminal. The reusable components provide the common services necessary for the user to coordinate both horizontally and vertically, with various communications and messaging systems, with any warfighter or organization within the Infosphere. The tactical information requirements are provided through data warehouses and common data standards that allow the user "to define his or her own Battlespace and to 'plug in,' 'pull,' or to have 'pushed' timely, relevant information anytime or anyplace."

The final tier, an expansion of the client-server data layer is the Infosphere. It represents a global web of interconnected military and commercial communications and information systems that will allow information access worldwide [Ref. 16:, page 10]. In addition, the Infosphere is envisioned as providing resource management and acting as a warehouse for C4I deployable modules [data warehouses].

In order to achieve the level of functionality, integration and interoperability envisioned by C4IFTW, the DII incorporates two unique and complementary strategies. The first is the DII COE, a system design and development framework aimed at providing commonality, interoperability, reusability, scalability, standardization and security.

> The DII COE model is analogous to the Microsoft Windows® paradigm. The idea is to provide a standard environment, a set of standard off-the-shelf components, and a set of programming standards that describe how to add new functionality to the environment. The Windows paradigm is

> one of "federation of systems" in that properly designed applications can coexist and operate in the same environment. But simple coexistence is not enough. It must be possible for applications to share data. The DII COE extends the Windows paradigm to allow for true "integration of systems" in that mission applications share data at the server level. [Ref. 6]

In addition, the DII COE encompasses architecture, shareable data, and automated integration [Ref. 6].

The second strategy creates a shared data environment; it is an extension of the DII COE. The SHADE program is seeking to establish a network of integrated databases within the COE accessible to decision makers worldwide. The SHADE is used to support the overall objectives of the DII by providing mechanisms for mission application integration, common data standards, common data access methods, and shared databases.

> The purpose of the Shared Data Environment (SHADE) is to facilitate data sharing and interoperability within the Defense Information Infrastructure (DII). The SHADE defines common services, tools and procedures to enable DII mission applications to maintain and share data. It will extend the DII Common Operating Environment (COE) by providing for the COE's data and database management requirements through the development and use of shared database segments and shared database servers. The objective is to integrate the data produced by separate applications and organizations into a global view of the battlespace and to make this integrated data available to users as required. [Ref. 4]

Based on the contributions of the DII COE and SHADE project, The DII encompasses an environment that is conducive to capability evolution. Evolving

capability refers to the inherent capability to add functionality, as needed without radical and costly changes to the infrastructure. The COE and SHADE technologies facilitate this capability by allowing applications developers and program managers a stable standards-based technical infrastructure in which to introduce new functionality.

3. Summary

The DII technical strategy and concepts are focused on creating a global environment of integrated information systems in line with the C4IFTW concept. The DII information infrastructure is an "open" distributed client-server environment. It facilitates application integration and a concept of "capability evolution" through data sharing and a common operating environment. Integrated functional data sets, in the form of data warehouses, provide flexibility and support for streamlined decision processes by providing access to timely relevant data. And finally, the communications systems are designed to provide ubiquitous connectivity and access to the network of distributed data centers, information services and resources, anytime, anywhere in the world.

D. IMPLICATIONS FOR PROCESS INNOVATION AND AGENT TECHNOLOGIES

This vision of computing will inevitably create an environment that facilitates the "rethinking," redefining, integrating, and streamlining of DoD processes, including the acquisition process. Within the acquisition system, the impact and implications of the DII are being considered with great thought and deliberation. Dr. Mestrovich, the DUSD

(A&T) Information Management Executive (IME) presented this vision of the future acquisition system, in a 1997 speech:

> The DII and GII [Global Information infrastructure] are both growing rapidly. Over the next several years, the information revolution will provide increased information access to all Government and Industry partners and expedite communication and information flow. This expanded electronic environment linking Government and Industry partners will be used to facilitate business transactions and conduct Electronic Commerce (EC). Linkages between Government and Industry partners will be made via multiple EC "paths", such as the Internet and Electronic Data Interchange (EDI) networks. The use of multiple EC paths will enhance the flexibility and agility of the information infrastructure and expedite business transactions.
>
> As we implement the Paperless Contracting initiative and continue to expand our electronic interfaces, business to business transactions will be increasingly conducted via this electronic buyer/supplier interface. The EC goal of the Department of Defense is to electronically connect all of its buyers with DoD suppliers of goods and services. Multiple types of electronic connections (EC paths) will become part of this EC infrastructure and will be used to link buyers and suppliers. The use of technology enablers (e.g., smart cards and web applications) to establish appropriate EC paths will maximize the flexibility of our buyers while ensuring a "level playing field" for our suppliers. Internally, DoD must also be able to electronically link our procurement, payment, logistics, and accounting systems to take full advantage of the information revolution and reduce our overhead costs and acquisition cycle time. [Ref. 26]

In addition, there are great implications for the use of agent technologies. Agent technologies can enhance the potential capabilities inherent in the DII by providing localized intelligence as needed throughout the enterprise. Agent technologies can

support intelligent search and presentation of heterogeneous data, intelligent advice and assistance, facilitate various forms of workflow, and provide automated event processing. The DII, based on its guiding design principles and technical strategy, offers a rich environment for business process innovation and agent technology integration.

1. Implications for Process Innovation

On a small scale, the implications for process innovation are seen in the efforts of the U.S. Patent and Trademark Office. The U.S. Patent and Trademark Office licensed an off-the-shelf procurement application to integrate procurement, finance and suppliers through electronic data interchange (EDI), create shared data resources and provide workflow automation [Ref. 7]. The result was an electronic workflow system that is credited with providing a paperless procurement process, reducing data entry errors and reducing decision cycle times.

> The work of the Procurement Office, the review and approval process, and the link to the vendor community have been electronically streamlined using an off-the-shelf system. Procurement speed and productivity have increased by creating a paperless process. Processing time from request to purchase order is down by 50%. ...For example, with EDI, RFQs [Request for Quotations] are routinely left open only 2 or 3 days instead of 2 weeks. [Ref. 7]

Truly, a computing environment that facilitates workflow and provides mechanisms for accessing shared data resources allows organizations to achieve dramatic productivity gains. In addition, as a consequence of the new automation capabilities and the productivity improvements, the organization was able to redirect resources and "rethink" their business processes.

> Procurement management has redesigned staff roles and taken advantage of the new technology to shift procurement personnel to higher value work and higher levels of professionalism. The clerical, tracking and paper burdens of the small purchase process have been eliminated. The system makes it possible and reasonable to delegate budget and procurement authority for small purchases to program offices, giving them more control over their own destiny, and freeing up procurement for high-end contracting. [Ref. 7]

It is exactly this type of "rethinking" of processes and innovation that DoD is seeking to achieve through their efforts to create the DII enterprise vision. As noted in our example, the effective integration of information technology and process reengineering provides a powerful tool for creating agile, productive and efficient organizations.

The dramatic transformation of the U.S. Patent and Trademark Office's procurement process provides an excellent example of what is possible throughout the DoD enterprise and specifically within the acquisition business function. Under acquisition reform, DoD is investing in just such enabling technologies, transforming processes and redefining statutory boundaries in an attempt to reduce costs, shorten procurement lead times, provide responsive services to the customer and transform the system into a world class buyer.

2. Implications for Agent Technologies

Software agent technologies can support those efforts by providing functionality and capabilities aligned with the goals and objectives of acquisition reform and process innovation. Although in its infancy, software agent technology is an exciting and quickly evolving software design and programming abstraction. It seeks to marry innovative

artificial intelligence techniques with sound software engineering methodologies and network technologies. The objective is to provide systems that are intelligent, adaptable, modular and capable of conducting delegated tasks within a distributed heterogeneous environment, such as the DII.

In [Ref. 27], an intelligent agent enterprise framework is presented that supports the use of intelligent agents within large, geographically dispersed environments such as the DII. The synergy between humans and agents is primarily accomplished through the software agent's use of an explicit machine executable representation of the enterprise. It contains a model and description of "personnel, facilities, equipment, inventory manufacturing processes and other corporate assets." [Ref. 27: page 1391] The knowledge representation of the enterprise contains the necessary knowledge for agents to perform tasks and coordinate with other agents to perform tasks.

The use of software agent technologies within the DII presents the same notions of machines as task owners and "virtual" assistants. This vision of enterprise computing is a significant shift from current notions of direct manipulation tools, where workers are the predominate centers of intelligence and required to initiate all actions. Software agents coupled with the inherent capabilities of the DII can provide an enhanced integrated environment supporting four fundamental enterprise-computing goals.

- **Information Access** - Facilitate easy access to enterprise-wide data resources across a distributed heterogeneous environment. Facilitate the fused presentation of multiple data sources at the desktop.

- **Monitoring and Automation** - Provide automated response and user notification of events within the agent domain.
- **Cooperative Work** - Facilitate the business process through task execution and interaction with other agents, other systems, legacy applications, and human agents.
- **System Integration** - Independently developed software must be easily integrated into the environment. The environment must support an incremental approach to extending capabilities and automating processes. [Ref. 27: page 1391]

The compulsion for intelligent agent use within the DII can be seen in the nature of the computing environment. The DII, as a technological innovation, is analogous to that of the World Wide Web (WWW) and integrated application suites in the commercial sector. It is similar in that it represents a large domain of information resources and provides an environment of integrated applications with significant functionality. Both the WWW and integrated applications have been tremendously successful technologies. However with their introduction a new set of unanticipated problems and challenges emerged.

For instance, the large unstructured nature of the WWW introduces an enormous search and resource location problem. The "browse" and "surf the Web" paradigm is unacceptable in terms of producing business efficiencies and is in most cases a distracter to normal business activities. Intermediaries, in the form of search engines, are being used facilitate information access by categorizing the WWW and thereby reducing the size and complexity of the search space.

The explosive growth in features and functionality, within modern integrated application suites, has made it impossible for the average user to understand or exploit the system's full capabilities. The introduction of "helper applications" such as "wizards," context sensitive help functions, templates, and assistant applications are provided in an attempt to reduce that complexity and guide the user through desired tasks. To that same end, software agents are tools that are seeking to provide a solution to some of the emerging complexity problems associated with large information rich environments, such as the WWW and complex feature rich integrated application environments.

The DII's standard client-server model married to the logical three-tiered C4IFTW architecture provides our technical framework for future agent implementations. Current agent technologies can provide utility at each layer within this model. On the DII desktop, we see software agent interfaces used to hide and manage complexity by advising the operator on policy, task execution, and system capabilities. In addition, agents can be effective in executing routine or time consuming information processing tasks, such as presenting customized information, setting appointments, filtering messages or conducting intelligent searches on the WWW [Ref. 8] [Ref. 33].

In the business function layer, agents provide a capability for integrating business rules, automating tasks, and integrating legacy systems [Ref. 20]. For example at the Massachusetts Institute of Technology (MIT), an agent-based system was created to support the buying and selling of goods within a virtual marketplace [Ref. 1]. The expected benefit of creating such a system is to "reduce transaction costs associated with

certain types of transactions (end-customer to end-customer transactions) where currently financial, time, and trust issues can impede negotiations and commerce."

In addition, researchers at the British Telecom (BT) laboratories have created an intelligent agent-based process management system (APMS) [Ref. 25]. The APMS system provides a decentralized agent-based alternative to the more traditional centralized workflow systems of today. A decentralized service model allows an approach to workflow that "empowers semi-autonomous groups to define how they will perform and manage tasks and processes." It essentially pushes the definition of work processes down to the workgroup level allowing more flexibility, agility and responsiveness within the organization. The current APMS is capable of managing over 100 business tasks within the BT enterprise.

Within the data layer, agent technologies can intelligently search large databases and notify users of events, patterns and trends [Ref. 22], supplanting costly and timely off-line data analysis tasks. The notification of patterns and events within large data sets has immense implications for creating streamlined man-out-of-the-loop analytical processes in areas such as imaging.

E. SUMMARY

The DII represents the future of DoD global enterprise computing. It is a technical infrastructure that facilitates software reuse, interoperability, integration and information access. By providing data access and integration mechanisms, the DII clearly provides opportunities for process automation and innovation, through data sharing and

the integration of functional business processes. Its client-server architecture promotes access to distributed services and facilitates the decentralization of business processes creating flexibility and agility within the enterprise.

Agent technologies enhance the inherent capabilities of the DII by providing intelligent processing at each logical layer within the enterprise. An integrated agent-based DII environment can provide intelligent access to data, automated and collaborative task execution, and intelligent event notification through monitoring and analysis of large distributed databases. Software agents are vehicles used to manage complexity through the delegation of tasks and the assignment of responsibilities to intelligent autonomous software systems. Based on anticipated advances in software agents, it is not unreasonable to expect agent-enabled systems to provide dramatic productivity improvements on the order sought through reengineering and radical process redesign [Ref. 25].

III. AGENT CONCEPTS AND TECHNOLOGIES

> What we today call "agent-based interfaces" will emerge as the dominant means by which computers and people talk with one another.
>
> Nicholas Negroponte MIT [Ref. 23: page 102]

A. INTRODUCTION

Today, through the integration of Artificial Intelligence (AI) techniques, networking technologies and sound software engineering principles, research into software agent technologies is elevating and redefining our current notions of the role of automated systems within the business enterprise. The term "software agent" in its ultimate sense defines a computing metaphor, a computing metaphor reaching beyond current notions of computing and positioning our thinking of human-computer interaction well into the future. Its acceptance represents our desire for automated systems that are more collaborative, intelligent and task oriented than current technologies. It is a manifestation of our desire for systems that are more than just simple direct manipulation tools, but systems that are collaborative, user friendly and intelligent. The combination of these somewhat nebulous terms inspires a notion of computing similar to that seen on the Star Trek television series. Of course, this type of collaborative computing capability is still years in the future. But, it is toward this lofty goal that the agent computing metaphor and agent technology research is leading.

This chapter presents the micro-view mentioned above, as it explores some of the basic concepts and technologies associated with software agents. Our primary focus is to define software agents in the context of this research and provide an indication of the state of current agent technologies.

B. BACKGROUND

Although there is no lack of research in the area of software agents, the fundamental question of what defines a software system as an agent has not been clearly defined. In general, software agent technologies are a product of a number of converging disciplines of computer science.

One thread in the evolution of software agents can be traced to object-oriented design and programming. Citing work by G. Agha, Michael Wooldridge [Ref. 40: pages 26-37] writes "Of course, both the idea of an agent and the idea of an object owe a great deal to the pioneering work of [Carl] Hewitt on open systems and the ACTOR model."

The ACTOR model is a model of computing that is closely related to multi-agent systems in that it relies on asynchronous communications and is based on a concept of behavior-based objects interacting through message passing. Each Actor or object within this system will execute an internal state change or "replacement behavior" based on a message being passed to it. In multi-agent systems, autonomous agents interact asynchronously through message passing or based on an agent communications language (ACL), in order to complete tasks or solve problems.

Another thread in the evolution of agent technologies is rooted in the fields of AI, Distributed Artificial Intelligence (DAI) and the Human Computer Interface (HCI) community. The AI community is concerned with creating systems and models that emulate human thinking or behavior. In very crude terms AI is concerned with building "smart" systems. From the AI community, software agents are the beneficiaries of various problem-solving architectures and knowledge representation techniques. The DAI community is concerned with the dispersion of processing and intelligence. As it pertains to multi-agent systems (MAS), DAI research addresses issues relating to task decomposition, control mechanisms, agent coordination, cooperation and solution synthesis. The field of HCI is making advances in terms of how humans interact with computers. In an Internet Computing article, Pattie Maes states, concerning the goal of HCI and agents, that:

> At the MIT Media Lab's Software Agents Group, we're trying to change the nature of human-computer interaction. I personally believe more proactive and more personalized software is of crucial importance because our computing environments are becoming more and more complex, and we as users can no longer stay on top of things. [Ref. 13: page 11]

Pattie Maes makes it clear that the objective is to create software interfaces that can provide greater utility and hide the complexities of the computing environment.

Each thread contributing to the evolution of software agent technologies provides a necessary component in providing intelligent, adaptable, and accessible automated systems. Each research community and commercial enterprise, pursuing theories of agency and developing agent technologies, is doing so from a different motivation and

intellectual frame of reference. Consequently, a singular conceptual or technological model defining a "software system" as an agent is not available.

C. DEFINING AGENCY

> The problem is that although the term is widely used, by many people working in closely related areas, it defies attempts to produce a single universally accepted definition. This need not necessarily be a problem: after all, if many people are successfully developing interesting and useful applications, then it hardly matters that they do not agree on potentially trivial terminology.
>
> Wooldbridge and Jennings [Ref. 39: page 4]

The research and development community at large has failed to form a consensus on a clear definition. And, we will not attempt to "muddy the waters" by offering another. However, we do need to introduce the agent concept in basic terms in order to specify how the term "software agent" will be used throughout this study.

In seeking to understand the concept of software agency, we use two complementary views. First, we infer agent attributes from the commonly used definition of the term "agent." Then, we use a more abstract approach to the term that is based on applying an intentional stance[2]. The intentional stance provides an abstraction and convenient means for describing complex system behavior. Following this introduction to the agent concept, we dictate our position on agency.

---

[2] The intentional stance allows humans to describe the behavior of a system in terms such as belief, desire, intention or some other such human mental attribute [Ref. 38: page 8].

1. By Definition

The term software agent is derived from the human concept of an agent. According to the American Heritage Dictionary [Ref. 34], an Agent is "*One that acts or has the authority to act; one that acts for or as the representative of another.*" This basic definition provides a wealth of useful concepts in understanding the basic definitions and attributes of (software) agency. First, the agent is an individual entity. The term individual implies a single entity with the capacity to act independently or *autonomously.* Second, the definition indicates that an agent represents another. In representing another, an agent must be directed by understanding or knowing the desires of the one being represented. Therefore, agents must possess *knowledge* and be *goal-oriented* toward the represented parties' desires. And finally, an agent *acts.* This implies the agent has a means of interacting with the environment. This interaction involves sensing, processing and effecting. Sensing simply involves monitoring or observing the environment. But, this can also imply *persistence* on the part of the agent. In order for the agent to process its environmental stimulus, it must posses some sort of *intelligence* or at the very least a rule based sequence of appropriate tasks or conditions that are goal-oriented and consistent with the represented parties' desires. Effecting implies a means to change the state of the environment. It can be accomplished in a number of ways but principally through *communicative acts* or sending messages. And possibly, the act of effecting may require the agent to physically go (*mobility*) to the place where it is needed.

This description points to a number of significant attributes consistent with varied definitions offered by the community at large. For example, in the work of Wooldbridge

and Jennings, they introduce a concept termed "*a weak notion of agency*" to define agency. It is based on the characteristic agent properties of autonomy, social ability, reactivity and pro-activeness. They are defined as follows:

> Autonomy: agents operate without the direct intervention of humans or others, and have some kind of control over their actions and internal state (Castelfranchi, 1995)
>
> Social ability: agents interact with other agents (and possibly humans) via some kind of agent communication language (Genesereth and Ketchpel, 1994)
>
> Reactivity: agents perceive their environment, (which may be the physical world, a user via a graphical user interface, a collection of other agents, the Internet, or perhaps all of these combined), and respond in a timely fashion to changes that occur in it
>
> Pro-activeness: agents do not simply act in response to their environment, they are able to exhibit goal-oriented behavior by taking the initiative. [Ref. 39: page 4]

These definitive attributes are strikingly similar to our description above. This suggests that an agent is autonomous, goal-oriented, knowledgeable, persistent, intelligent, communicative, action-oriented and possibly mobile. From a different perspective, we can see an equally valid notion of agency that is more aligned with how the system is perceived by the user or designer.

2. As an Abstraction

In Shoham's work, he provides this definition:

> An agent is an entity whose state is viewed as consisting of mental components such as beliefs, capabilities, choices, and commitments. These components are defined in a precise fashion, and stand in rough correspondence to their common sense counterparts. In this view, therefore, agenthood is in the mind of the programmer: what makes any hardware or software component an agent is precisely the fact that one has chosen to analyze and control it in these mental terms. [Ref. 31]

As pointed out by Watt [Ref. 37], the use of the term agent, agent oriented programming, and mental states, described by Shoham is metaphorical. And clearly from Shoham's own words, the terms agent and mental states are used to facilitate a certain utility in order to *"analyze and control"* the states, capabilities and complexities of the underlying system. Therefore in this case, the term "agent" is an abstraction ascribed to the system based on a need or desire to describe the complexities of the system in mental terms.

In the field of human-computer interaction, the term agent is used to refer to a system that takes a user's goals and acts on them. This simple definition can include everything from automatic spell checkers to Internet based information retrieval tools. Watt examines this definition of agency from a user perspective and concludes with the following statement:

> ...they are clearly agents; they are capable of acting on their own, or rather, their users treat them as if they are capable of acting on their own -- and this is what really matters. It is the behavior, the whole behavior, and nothing but the behavior that counts [Ref. 37: page 30]

In other words, an agent can be anything that the "user" believes offers the type of utility that would be expected from an agent, according to the natural use of the term. The term agent becomes a convenient abstraction for describing complex system behavior.

3. Research Definition of Agency

For our purposes, we combine both views of agency. We use both a definitional notion of agency and an abstract notion of agency. In our definition, an agent is a goal-directed autonomous entity that has the capability to sense its environment, decide on a course of action and act. In specifying a system or process as an agent, we apply two criteria. First, any system with the specified processing capabilities (e.g., sense, decide, act) must preserve the minimum essential quality of autonomy. Autonomy is defined as the ability of an entity to exhibit self-directed behavior. And second, the designation of the term "agent" is beneficial in terms of understanding or specifying the system's behavior.

Our definition encompasses everything from time-triggered processes or scripts to adaptable and learning knowledge-based systems. For instance, a time-triggered print process exhibits a certain autonomy in terms of executing at a user preferred time, but is there any real value in specifying it as an agent? On the other, hand a user may find it more natural to refer to a similar process as an agent if it is mobile, executes across the network, and notifies the user of a completed action. We have purposely adopted a very broad definition of agency, primarily because we are more concerned with the utility of agent technologies as opposed to specifying agent theories.

D. AGENT TECHNOLOGIES

Software agent technologies are founded on a fundamental belief, within the AI community, that machines can be created to exhibit "intelligent" behavior. Although intelligent behavior is only a part of the agent computing metaphor, it does provide the foundation for systems that are collaborative, knowledgeable, task oriented and accessible. The agent computing metaphor brings to mind an image of automated systems that can effectively respond to audible instructions like "computer show me the cost schedule performance trends for the National Missile Defense program over the last two years."

Agent technologies needed to support this level of human and computer interaction are being developed through agent research. The evolution of agent technologies will ultimately allow the user to interact on a higher linguistic abstraction then is currently available (i.e., natural language verses programming languages). It will provide systems that intelligently advise and assist, and create systems that communicate, coordinate, and negotiate with other agent-based systems and interact with non-agent systems in executing tasks.

In the following terse discussion of agent technologies, we discuss agent technologies most closely associated with achieving the agent computing capabilities indicated above. For discussion we place the enabling technologies of software agents into three broad classifications:

1) Agent Cognitive Architectures

2) Agent Knowledge Acquisition and Representation

3) Agent Coordination

Agent Cognitive Architectures, developed through AI research, provide the agent's decision-making, behavioral and intelligence capabilities. Agent Knowledge Acquisition and Representation provides a means of acquiring and representing machine executable knowledge. And, Agent Coordination provides a means of controlling distributed multi-agent systems.

1. Agent Cognitive Architectures

In [Ref. 39], Wooldridge and Jennnings discuss three basic agent architectures and we will limit our discussions to them. The basic types are deliberative, reactive and hybrid. For an extensive overview of specific architectures, the University of Michigan has a WWW resource that provides a good reference [Ref. 36]. This classification of agent architectures is based on categorizing them as a particular type of knowledge-based system.

The deliberative agent type is defined as:

> an agent or agent architecture to be one that contains an explicitly represented, symbolic model of the world, and in which decisions (for example about what actions to perform) are made via logical (or at least pseudo-logical) reasoning, based on a pattern matching and symbolic manipulation. [Ref. 39: page 24]

The deliberative or declarative agent indicates an architecture that assumes that an agent's knowledge can be "declared" prior to the agent running. The declarative agent is constructed based on the use of the three basic elements associated with knowledge based systems: 1) control, 2) inference, and 3) knowledge representation.

Deliberative Architecture

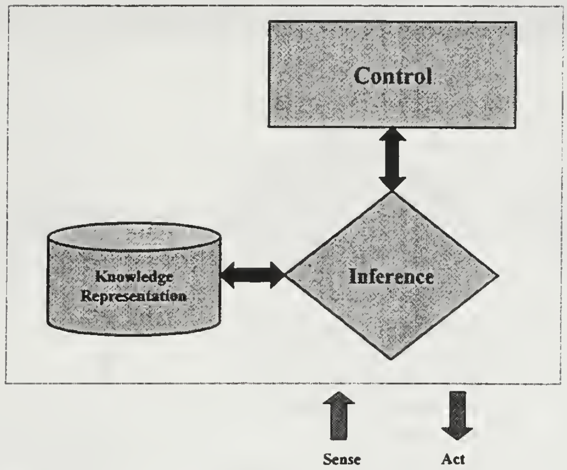


Figure 3: Deliberative Architecture

Control refers to the part of the architecture that controls the agent's reasoning processes. It is an approach to problem solving that tells the agent "how to reason" about a particular set of sensory inputs. The knowledge representation or knowledge base provides the agent with a machine executable logical representation of the problem space. We provide more detail on this important issue in the next section. And finally, declarative architectures have some sort of inference engine. The inference engine allows the agent to "infer" or draw conclusions about the validity of sensory events.

The technologies that support the declarative architecture are listed in Table 1. Our expectations are that in some way each will provide value in the continued pursuit of agent technological sophistication.

Knowledge Processing Technology Component	Current Capability	Emerging Capability
Representation	Rules, Frames, Classes, Hierarchies, Attributes, Propositions, Constraints, Demons, Procedures, Certainty Factors, Possibility Intervals, Fuzzy Variables	Domain modeling, Temporal spatial and Geometric modeling, Ontologies, Complementary
Inference	Forward Chaining, Back Chaining, Unification, Modus Ponens, Resolution, Inheritance, Hypothetical Reasoning, Incremental Reasoning, Simultaneous alternatives, Constraint propagation	Reasoning with very large knowledge bases
Control	Goal-directed, Data-directed, Message-passing, Demons, Focus of attention, Agenda, Metaplans, Triggers, Knowledge source instantiations, Expected values of pending actions, Scheduling	Distributed, concurrent and real-time control systems

Table 1 : Knowledge-Based Systems Component Technologies [Ref. 11: page 32]

In contrast to the deliberative agent, the reactive agent architecture does not use an internal symbolic representation of the world. It uses procedural logic and/or multi-layered behavioral models to select the appropriate response to sensory events. This is different from the declarative agent paradigm in that it assumes that intelligence is not based on an internal reasoning capability but an external demonstrated "intelligent" behavior.

Reactive Architecture

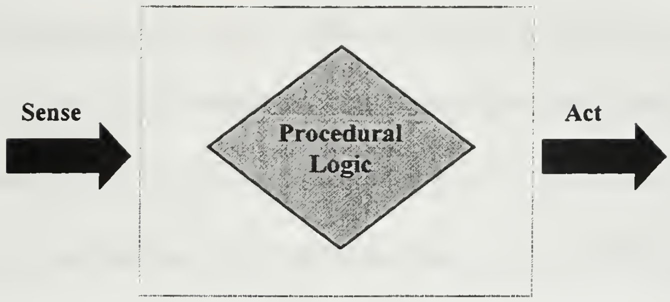


Figure 4: Reactive Architecture

The reactive agent does not reason about the sensory event, but simply "reacts" in an automatic fashion. This architecture is completely situation specific and does not provide the flexibility or generality of a declarative system. But, we would expect a performance advantage over declarative systems since declarative systems must execute reasoning processes for each event.

Finally, the hybrid system has the best of both architectures. It combines the generality and flexibility of the declarative system and the automatic response functions and performance advantages of the reactive system. In terms of classical knowledge based systems, most systems fall into the hybrid category [Ref. 29].

The current state of agent cognitive architectures is summed up below in the following statement: "A general-purpose agent system or an agent that has general knowledge about the world and can reason, learn and act within a broad range of environments is not possible with current technologies." [Ref. 30: page 843] Therefore,

current agent system architectures are constrained to using specifically defined procedural, behavioral or explicit knowledge in executing narrowly defined tasks.

2. Agent Knowledge Acquisition and Representation

Agent knowledge acquisition and representation specifies how knowledge is acquired and internally represented for execution. Knowledge acquisition and representation is traditionally called knowledge engineering and is principally associated with the process of eliciting, representing and creating a knowledge base for expert systems. As indicated in Table 1, numerous techniques are currently in use to represent and process various forms of knowledge. Knowledge engineering has been very successful in creating methods and representations for machine executable knowledge.

> I believe that knowledge engineering always works if you satisfy the applicability conditions. That is, if you can find somebody who does a reasoning task, does it repeatedly, and does it well, you can ultimately elicit this knowledge, represent it, and implement it in a machine. [Ref. 11: pages 103-104]

But beyond the fundamental requirements of knowledge engineering, knowledge acquisition in agent technologies is focused on acquiring knowledge through machine learning techniques. An unstated goal of agent-based systems research is to limit the amount of knowledge engineering tasks required in providing an effective system. Agent technologies are seeking to move beyond the costly restrictions imposed by knowledge engineering requirements and forge toward methodologies and technologies that more closely mimic the human knowledge acquisition process [Ref. 28: pages 546-555].

Humans acquire knowledge through sensor activities such as reading or listening. So we expand the use of knowledge acquisition to include the ability of an agent system to acquire new knowledge through sensory learning.

As indicated in [Ref. 14], techniques that use heuristics classification, neural networks, reinforcement learning, learning by observation, instructional learning, and case-based learning are all forms of machine learning being investigated for agent technologies.

> Of the available techniques, it is felt that the traditional classifier techniques are unsuitable in this domain which is basically unsupervised in nature. The unsupervised neural networks might have a role in classifying users if some sort of stereotypical system is employed. This being the case, we are left with a problem as the other techniques discussed above are either new and untested or very slow and thus probably unsuitable for a real time system such as an intelligent user interface. [Ref. 14: page 8]

With machine learning, agents are able to learn about their environment, the nature of the task and decide on appropriate actions. Learning is essential in most real-world environments. But based on the complexity of most real-world tasks and environments a certain amount of uncertainty is inevitable. In addition, the ignorance of the system designer or programmer, concerning the domain, contributes to the uncertainty of the system meeting its intended goals. So in order for an autonomous system to be truly effective, it must be able to adapt to unknown events and increase its performance over time.

In his book, *Being Digital,* Nicholas Negroponte states, concerning an intelligent human-computer interface, that:

> The idea of building this kind of functionality into a computer until recently was a dream so far out of reach that the concept was not taken seriously. ...The idea is to build computer surrogates (Agents) that posses a body of knowledge both about something (a process, a field of interest, a way of doing something) and about you in relation to that something (your taste, your inclinations, your acquaintances). [Ref. 23: page 151]

A recent study provides an evaluation of current learning techniques applied to meeting the goals of having knowledge about "something" and about the user "in relation to that something."

> Two main application domains stand out amongst the available literature. First, many systems exist which attempt an information filtering/retrieval role. These systems would appear to be quite successful in what they do, particularly the ones based on ACF [Automated Collaborative Filtering] clustering techniques. The ones based upon user models suffer from the fact that there doesn't exist a reliable way to extract meaning from arbitrary natural language text and therefore instead utilize keyword extraction. There also exists a role for the assistant style of agent for complex systems particularly those based upon multi-agent systems. ...Whatever the application, some degree of adaptivity is desirable in order to maximize the improvement in user productivity that the assistant is able to generate. Unfortunately most current agent based systems make little use of significant user modeling relying for the most part on quite simplistic user profiles. It would seem therefore that there is a need for more research into the use of user modeling in intelligent interface agents. [Ref. 14: page 13]

Therefore, it seems that current systems can learn about information spaces and filter or retrieve information based on user profiles, but to learn about the user directly evidently presents difficult and challenging problems that are unresolved.

3. Agent Coordination

But let us assume for moment that we build software agents with the characteristic attributes (e.g., autonomous, goal-oriented, intelligent) previously defined. Then the next question becomes, how do we get multiple agents to work together to perform complex tasks? In human endeavors, we routinely communicate, collaborate, and negotiate with others in order to complete tasks, share goals or resolve disputes. Although routine for humans, multi-agent coordination is a tremendously complex problem.

In fact, multi-agent systems are primarily limited by the need to control and coordinate the activities of autonomous, adaptable processing entities. The idea of multi-agent systems borrows its theoretical foundations from DAI and distributed processing research. The basic idea is that multiple agents, each with a limited area of expertise can be controlled and made to coordinate activities and cooperate to accomplish complex tasks too great for a single agent. In doing so, the benefits of multi-agent system coordination are:

- Maintain order; prevent chaos
- Ability to meet global constraints beyond the capability of a single agent
- Maximize utility through resource, expertise and information sharing
- Exploit group efficiencies through a team approach [Ref. 24]

In [Ref. 24], a number of techniques to achieve multi-agent coordination are reviewed and placed into four broad categories: 1) Organizational Structuring 2) Contracting 3) Multi-agent Planning 4) Negotiation. The results are summarized below:

1. Organizational Structuring - The simplest of the techniques. It defines a pre-activation structure that dictates the behavior of assigned agents. The two techniques noted are the Master/Slave and the blackboard. In the Master/Slave system one agent, the master, with full autonomy and domain knowledge, controls the activities of the slaves. The slaves have limited autonomy and specific task knowledge and respond to the commands of the Master. This approach often limits the performance expectations and other benefits of distributed systems. The Blackboard system as the name implies is basically a shared memory space for intermediate results. Each agent is allowed controlled access to the space to either read or write. The main disadvantage of blackboard systems is the potential for bottlenecks as the number of agents accessing the blackboard increases.

2. Contracting - Based on an open market model, software agents request and accept bids for required services based on the agent's capabilities. Each agent acts as both manager and contractor, the agent has the role of manager when requiring services and contractor when providing services. This approach is best used when the "task has a well defined hierarchical nature, the problem has a coarse grained decomposition and there is minimal coupling among tasks." [Ref. 24: page 47] It provides the following advantages: 1) dynamic task allocation, 2) dynamic reconfiguration of participating agents, and 3) load balancing. However, it does not support agents with competitive behaviors where conflict resolution is required. It also is communications intensive, which may be prohibitive in certain environments.

3. Multi-agent Planning - Multi-agent planning has two type centralized and decentralized. Centralized multi-agent planning uses a coordination agent to receive and synchronize, through conflict resolution the local plans of all agents involved in task execution. The coordination agent then creates and publishes an executable global plan. In decentralized multi-agent planning, the agent group will share local plans and communicate until a global plan is created. Each agent must build and modify its local plan based on communications and conflicts with other group agents. Multi-agent planning is limited in that processing and communications requirements are high and since coordination is gradual it may have limited applicability.

4. Negotiation - Negotiation techniques are the most widely researched area in agent coordination research. The use of negotiation requires agents to reason

> about the beliefs, desires and intentions of other agents. Negotiation techniques are classified under game theory based, plan-based and miscellaneous categories. Game theory based techniques use utility maximizing agents interacting within the constraints of a known utility payoff matrix. Each offer and counteroffer is evaluated based on the expected utility and the individual agent's negotiation strategy. This technique is limited based on the assumptions of full rationality, knowledge about the competitor's preferences through the payoff matrix, dual party negotiations and homogeneous non-adapting agent architectures. In each case, this does not seriously reflect the conditions of a real world negotiation. Plan-based techniques are currently ill defined and suffer from the same limitations as centralized and decentralized multi-agent planning techniques discussed above. Miscellaneous techniques such as logic, Case-based Reasoning (CBR), and constraint-directed. For information on these techniques please see the reference below.
>
> [Ref. 24]

The authors in [Ref. 24] conclude with the following lessons learned based on their review of agent coordination literature:

> ❖ One-off coordination strategies (or combinations of strategies) are devised and used in one-off projects. Hence, solid conclusions as to their scope, applicability, usability, etc, have not been established.
>
> ❖ There is little empirical or theoretical support for any strategy or strategies. Not enough studies have been done to validate many of the proposals.
>
> ❖ It is not very clear when, where, how and why various negotiation and coordination strategies or combinations of them are used in various applications or proposals.
>
> ❖ The contract net and the master/slave models and variations of them appear to be the most used strategies due to their simplicity. ...
>
> ❖ Most coordination or negotiation strategies do not involve any complex meta-reasoning required of most domains. ...
>
> ❖ There is a lack of fundamental analysis of the process of coordination and negotiation.
>
> [Ref. 24]

The lack of clearly defined coordination strategies limits the deployment of large-scale systems within the enterprise. The development of standard coordination models and strategies are critical for achieving software agent technological maturation.

E. AGENT DEVELOPMENT TOOLS

An important area defining the maturity and utility of software systems is the availability and quality of development tools. A survey of a number of agent development tools is recorded in table 2.

Product	Company	Languag	Description
AgentBuilder®	Reticular Systems, Inc.	Java	Integrated Agent and Agency Development Environment
AgenTalk	NTT Japan Ishida Labs		Agent Coordination Protocol and Description Language
ABE	IBM	C++, Java	Agent Development Environment
Aglets	IBM Japan	Java	Mobile Agents
Concordia	Mitsubishi Electric	Java	Mobile Agents
Intelligent Agent Library	Bits & Pixels	Java	Agent Library
Kafka	Fujitsu	Java	Agent Library
LiveAgent	AgentSoft Ltd.	Java	Internet Agent Construction
Microsoft Agent	Microsoft Corporation	Active X	Interface creatures
Odyssey	General Magic	Java	Mobile Agents
Voyager	Object Space	Java	Agent-Enhanced ORB

Table 2 : Agent Development Tools [Ref. 12]

A more extensive listing of agent development tools is provided in Appendix B. The lack of an accepted software agent definition, standard architectures, standard agent communication languages or standard coordination mechanisms prohibits a serious evaluation of the true state of agent development tools. So we are content with simply informing the reader on the availability of such tools.

F. CHAPTER SUMMARY

The maturation of agent technologies is critical to achieving any significant gains in utility or user productivity using the agent paradigm. Advances in the area of machine learning, cognitive architectures and multi-agent coordination are critical. Current technologies are capable of executing clearly defined user tasks or greater complexity tasks within narrowly defined problem spaces, but an agent operating within a broad problem space using general and common sense knowledge is unavailable.

As the technology matures the expected utility of software agents is seen in two areas. First, agent technologies are attempting to narrow the gap between the way humans interact and communicate and the way machines communicate. The ability of applications to better understand natural language (e.g., verbal and written) constructs and effectively act on them creates a more natural and productive interaction between man and machine. This narrowing will allow users to effectively communicate their tasks, goals and desires to the application and have the application learn, adapt and execute based on that transfer of knowledge. This is clearly a substantial gain in terms of the usability of automated systems.

Second, agent technologies provide utility in terms of functional capability. Intelligent or pseudo-intelligent systems that are subject to delegation provide the advantage of reducing the cost of certain activities that are time consuming, complex or repetitive. Agents acting as user surrogates searching the web, negotiating simple purchases or monitoring events and other processes, based on the user's stated desires, are clearly productivity enhancing.

IV. SOFTWARE AGENTS AND THE DOD PROCUREMENT ENTERPRISE

A. INTRODUCTION

Information technology (IT), applied to the business enterprise, is heralded as the impetus behind modern process reengineering successes. It creates a flexible virtual environment for task execution and collaboration that is temporally and spatially independent. Workers are no longer tied to a physical location or a particular set of work hours to provide value to the enterprise. It also provides tools for task automation, event monitoring, information processing, and information access and distribution. Information technology, applied to business processes, creates efficiencies and productivity gains allowing for personnel reductions, process innovation and process redesign, in conjunction with the decentralization of work processes.

In this chapter, we use agent-based systems, as an advanced IT enabler, for redesigning processes within the DII Acquisition system. We use the Simplified Acquisition Procedures (SAP) as the focus of our redesign efforts. The SAP is specified in the Federal Acquisition Regulation (FAR) and represents a key element of acquisition reform. To accomplish this task, we represent the process using a traditional process-flow model, employ Use Case analysis to integrate the DII macro-process view and the software agent micro-technology view, and use a heuristic measure of process complexity to identify processes suitable for machine verses human performance.

By exploiting the inherent strengths of both software and human agents, we seek to enhance productivity by freeing human agents from routine tasks and enabling the refocusing of human resources to high value acquisitions. In the redesigned process, agent-based systems have process ownership roles much like that of an employee within the enterprise. The result is an agent-based redesign of SAP processes where human agents and software agents share in the responsibilities for process execution.

B. PROCESS REPRESENTATION

The SAP was selected for a number of reasons. As the name implies, it is by far the least complex of all the DoD procurement processes. Yet the SAP still involves all essential elements of defense procurement. The SAP is also more aligned with commercial buying practices and this allows greater access to commercial products and small business. Finally, the SAP allows us to study interactions between the three principal organizations within the DoD acquisition process: the buying agency, the contracting agency and the vendor.

The traditional process flow diagram, based on task decomposition, provides a suitable representation of the process domain and a means of introducing the SAP for discussion.

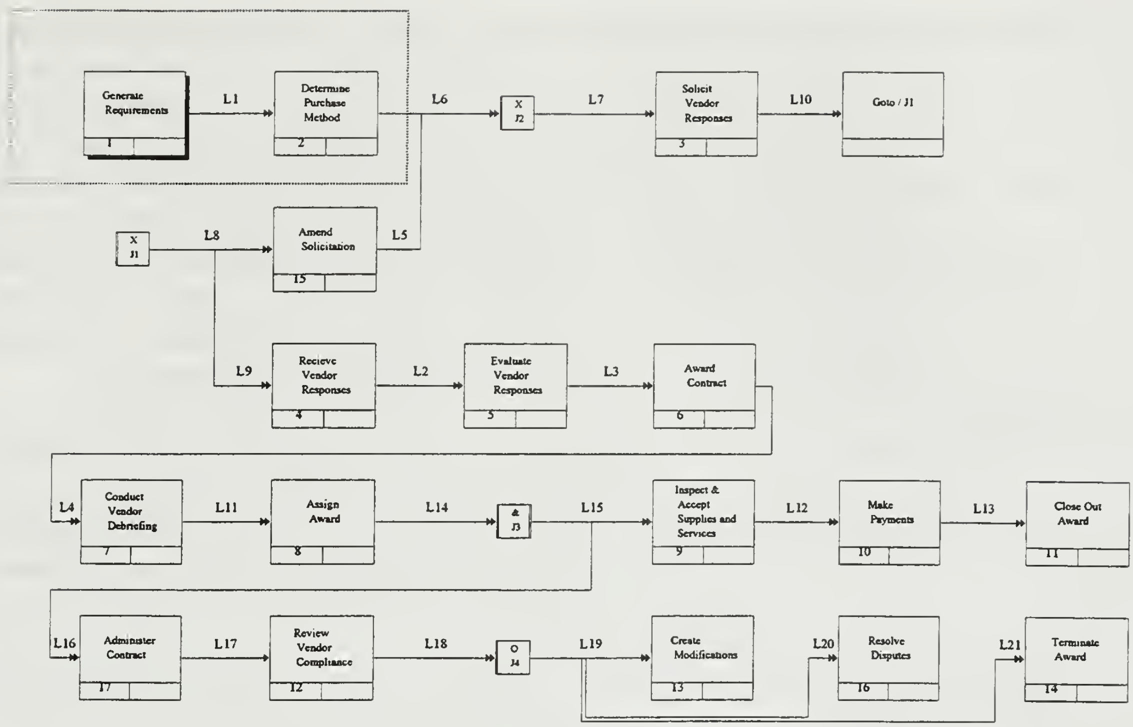


Figure 5: Simplified Acquisition Procedures Process Representation

This SAP process flow diagram looks similar to the process flows for most classes of DoD procurement. But the SAP is unique because of its simplified scope of activities and reduced bureaucratic overhead. The SAP procurement process has three principal phases: 1) Requirements, 2) Pre-award, and 3) Post-award. Each phase is briefly summarized below:

1.) Requirements Phase - Requirements are created and documented by the buying agency. The Requirements Generation task, in conjunction with the Determine Purchase Method task, specifies the type and format of the buying agency description of need (e.g., requirements document). The output of this phase is the requirements document. It must contain a suitable description of the requirement in order for vendors to clearly understand the needs of the Government [Ref. 3: part 12.202 (b)].

2.) Pre-Award Phase - The requirements document is translated into a solicitation, by the contracting agency and is conveyed to eligible vendors for consideration and responses. Based on the method of purchase and the specifications of the solicitation, offers are received, validated and evaluated

for contract award. The result of the evaluation process is the selection of the best responsive offer from a responsible vendor. The tasks associated with this phase are Solicit Vendor Responses, Receive Vendor Responses, Evaluate Vendor Responses and Award Contract (e.g., Task Order, Delivery Order, Blanket Purchase Agreement call, Purchase Order, Government Purchase Card Purchase, Solicitation-Offer-Award for Commercial Items). The term "contract" is generally used to indicate a valid Government contractual obligation in whatever form [Ref. 3: part 2.101]. The one exception case to this phase is the Amend Solicitation task. It is performed when the initial solicitation must be modified for whatever reason.

3.) Post-Award Phase - The contract is administered and the vendor's compliance is reviewed in accordance with the stated provisions of the contract, the FAR and FAR supplements. After awarding the contract, losing vendors are notified and debriefed. The normal execution of the process is where the vendor is in compliance, supplies and services are received and inspected, payments are made and the contract is closed out without any exception tasks being executed. The tasks executed are Conduct Vendor Debriefing, Assign Award, Inspect & Accept Supplies and Services, Make Payments and Close Out Award. In administering the contract there are a number of exception cases associated with reviewing the vendor's compliance. They are Create Modifications, Resolve Disputes and Terminate Award.

Although the process flow diagram provides an excellent vehicle for introducing the general process, it does not delineate the process domain with sufficient fidelity for our ultimate purposes. Our intent is to clearly delineate the process domain and disaggregate the SAP to a more manageable level for analysis.

We do this by applying process specialization to the process flow diagram. The process flow diagram above is created based on applying task decomposition and presents the process as a sequence of individual tasks. Process specialization, on the other hand, focuses on the interrelationships between tasks and the restrictions imposed upon each specialized process by the FAR. It presents a view of the process domain that identifies

aggregates of tasks that are related in knowledge requirements, control mechanisms and interactions.

The SAP process has a number of specializations documented in Chapters 12 and 13 of the FAR. The execution of each specialization is conditioned on a particular set of inputs related to a specific procurement.

- Acquisition of Commercial items - Policies and practices instituted in the FAR more closely resembling those of the commercial marketplace. [Ref. 3: part 12]

- Delivery Orders - An order for supplies placed against an established contract or with Government sources. [Ref. 3: part 2.101]

- Micro-Purchase (Government Purchase Card) - An acquisition of supplies or services (except construction), the aggregate amount of which does not exceed $2,500, except that in the case of construction, the limit is $2,000. [Ref. 3: part 2.101]

- Micro-Purchase (Imprest Funds) A cash transaction for the acquisition of supplies or services, the aggregate amount of which does not exceed $500 or such other limits as have been approved by the agency head. [Ref. 3: part 13.403]

- Task Order - An order for services placed against an established contract or with Government sources. [Ref. 3: part 2.101]

- Indefinite Delivery / Indefinite Quantity (ID/IQ) - A contract for supplies or services that does not procure or specify a firm quantity of services (other than a minimum or maximum quantity) and that provides for the issuance of orders for the performance of tasks during the period of the contract. It provides flexibility in both quantities and delivery scheduling; and ordering of supplies or services after requirements materialize. Indefinite-quantity contracts limit the Government's obligation to the minimum quantity specified in the contract funds. [Ref. 3: part 16.501-1]

- Blanket Purchase Agreement - A Blanket Purchase Agreement is a simplified method of filling anticipated repetitive needs for supplies or services by establishing "charge accounts" with qualified sources of supply. [Ref. 3: part 13.201]

The Task/Delivery Order, ID/IQ Contract and the Blanket Purchase Agreement are pre-existing contracts available for meeting the needs of the buying agencies. A potential buyer need only identify the items needed and execute the order based on the ordering procedures of the contract. This approach is the simplest and most timely for meeting buying agency requirements for purchases over the micro-purchase threshold.

Using our SAP specialization to disaggregate the process domain further, the SAP tasks and specializations are now visible as depicted in Figure 6.

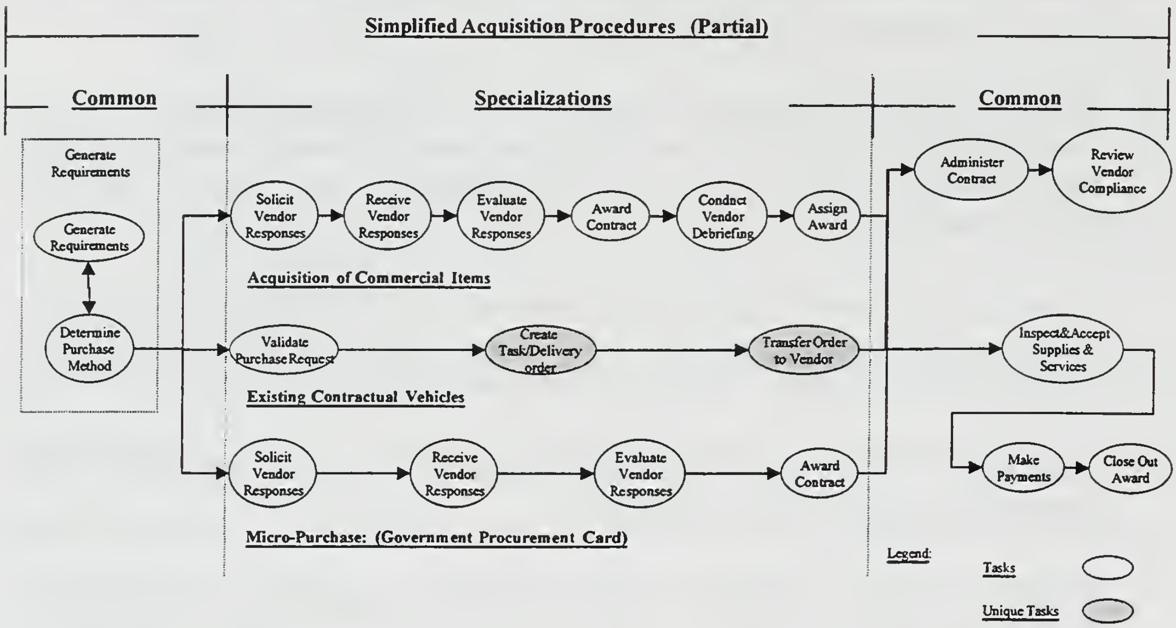


Figure 6: SAP Process Flow Model

We combine the pre-existing contract methods (e.g., task/delivery order, ID/IQ, blanket purchase agreement) for clarity into one category and remove the Micro-purchase (Imprest funds) specialization, because the latter requires cash transactions and we are concerned with executing tasks within a virtual environment. The figure delineates three basic specializations - acquisition of commercial items, existing contractual vehicles and

micro-purchase (Government Purchase Card) - and differentiates between common and unique tasks. Common tasks are shared by all specializations with minimal variation across the process domain, tasks such as ensuring vendor compliance through contract administration. For simplified acquisition there is typically no need to have Government personnel at the contractor's facility, therefore the administration functions are normally in-house and consistent regardless of specialization. The Make Payments task is executed by the Defense Finance and Accounting Service (DFAS) based on verification of contract compliance and the submission of the vendor's invoice. Unique tasks, on the other hand, are a part of a specific specialization. In the case of existing contracts or agreements, the submission of a task/delivery/call order represents an obligation to the Government, as opposed to the basic contract being the obligation document. With the specializations and tasks further delineated within the diagram, we are now at a manageable level and can use Figure 6 in illustrating our concept and supporting analysis.

We concentrate on the Acquisition of Commercial Items specialization for analysis in this chapter. It provides an informative and representative specialization of the process domain yet it generalizes well (e.g., through the solicitation-offer-evaluate-award process flow) to most DoD procurements. In this study, the process task is the level of analysis. Each task is a representation of a number of human decision processes and has a specific task environment that influences its execution. Therefore the process tasks associated with the Acquisition of Commercial Items specialization are the subject of analysis in the following sections.

C. THE INTEGRATION OF THE MACRO AND MICRO VIEWS

Agent-based process redesign requires that we integrate the micro-technology view of agent technologies into the macro-process view of the DII SAP process domain. By presenting a model that accommodates both views, we analyze the domain using the same heuristic and conceptual model. The key to this approach is the decision node concept. The decision node is defined as "a point [node] in the process where some local intelligence, or for that purpose autonomous, 'processing' is necessary to ensure validity of the undertaken action." [Ref. 35: page 584] We use decision nodes to identify tasks, within the process domain, that encapsulate process intelligence, control and knowledge.

In order to integrate the macro and micro views, we use an Object Management Group[3] (OMG) design artifact called the Use Case diagram. It is created through identification of agents, objects and decision nodes [tasks] within the process domain. The Use Case diagram is a visual artifact of the Unified Modeling Language, a language developed for specifying, visualizing, constructing, and documenting the artifacts of software systems. The diagram maps the agent's (i.e., human agent or software agent) interaction with the system through decision nodes that identify how the system is "used." The human decision nodes [tasks] are represented as "use" functions. In Chapter III our definition of an agent, from a micro-technology view, was "a goal-directed autonomous entity that has the capability to sense its environment, decide on a course of action and

[3] Object Management Group - A standards-setting consortium of over 300 organizations aimed at establishing standards for object-oriented and distributed object technologies.

act." It is in essence a realization of a human decision process. It is this definition of agency coupled with the Use Case diagram that integrates the macro and micro views.

The software agent as a realization of a decision process provides a means to supplant human decision nodes with agent technologies within the Use Case diagram.

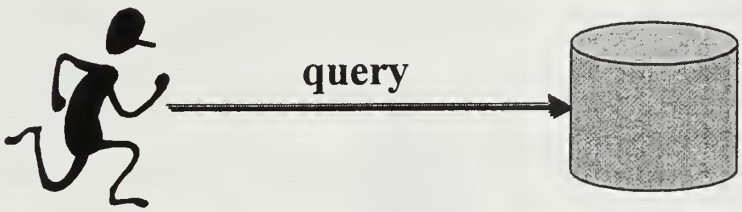


Figure 7: Simple Use Case Example

In the simple example characterized in Figure 7, a human agent must query a local SHADE database. This Use Case diagram identifies three entities within the process domain: the agent, the "query" decision node and the SHADE database. The "query" decision node represents the human agent's task relationship with the database and the decision processes necessary for interaction. It is also the focus of our analysis as a potential agent candidate. The decision node [query] contains the process intelligence, procedural knowledge and the control structures necessary for executing the task. If the task is suitable for agent incorporation the human agent can be replaced with a software agent for task execution. From this simple example, we see that the Use Case diagram offers a powerful abstraction for visualizing the process domain and integrating the macro-process view of the DII and the micro-technology view of agent technologies.

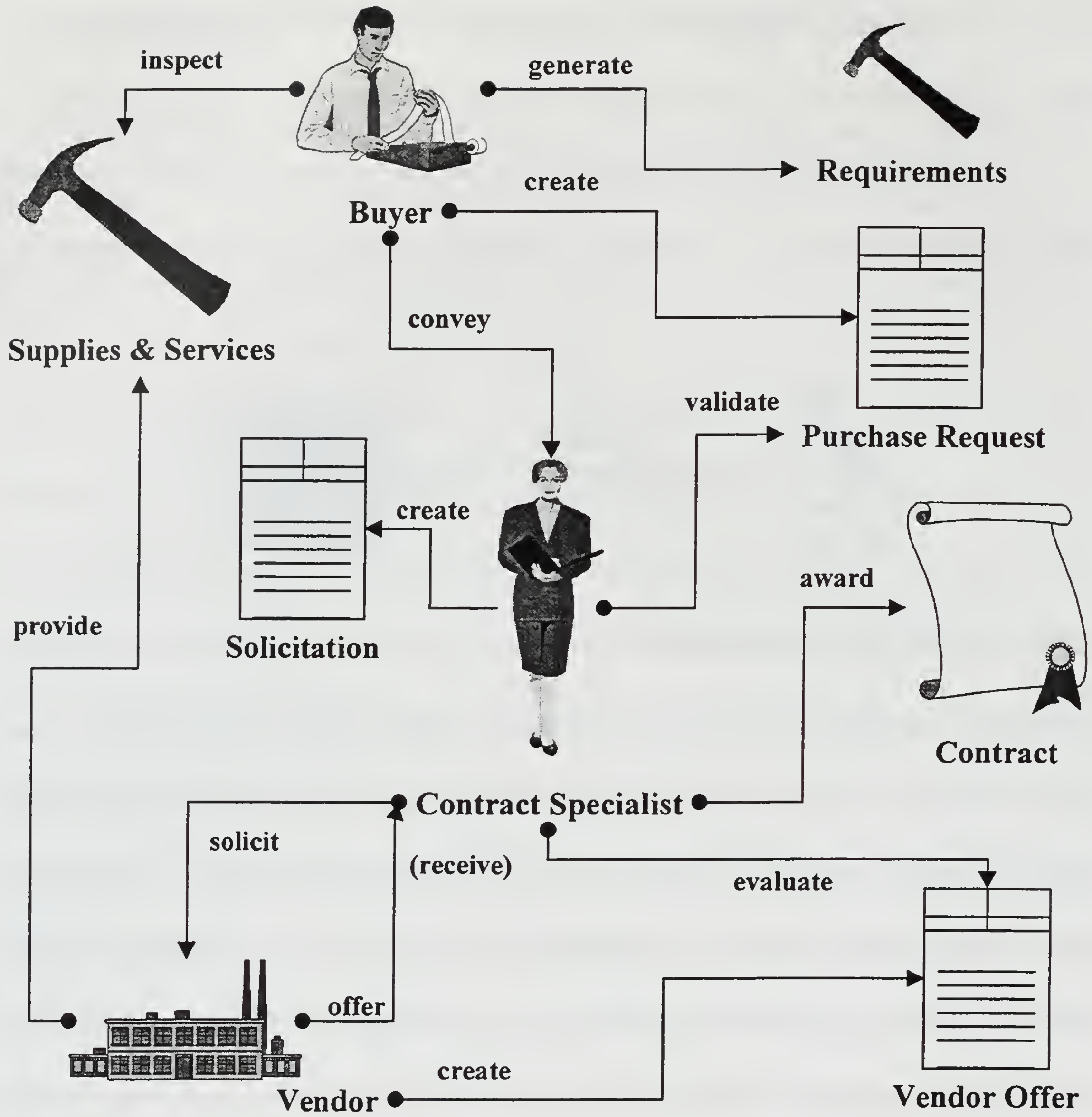


Figure 8: Use Case: SAP Acquisition of Commercial Items (partial)

Figure 8 provides a (partial) Use Case representation of the Acquisition of Commercial Items specialization. This Use Case diagram represents the primary tasks executed by the buyer, contract specialist and vendor in awarding a contract for commercial items using the SAP process. It is assumed that a paperless acquisition process is the standard and that the DII infrastructure is in place to support access to

acquisition related information resources, the presentation of data and the interconnection of agents. At the top of the diagram, the buyer initiates the process by generating requirements, a complex search and data analysis, fusion and synthesis task. It results in the information necessary to create a valid purchase request. The purchase request is conveyed to the contract specialist for an internal validation at the contracting agency and is translated and fused with contractual policy requirements into a valid solicitation. The solicitation is conveyed to eligible vendors that create and reply with an offer to the contract specialist. The offer is deemed responsive or non-responsive and is evaluated based on the stipulations of the solicitation. The best responsive and responsible vendor is awarded the contract, based on the reasonableness of the price and possibly other factors. The supplies/services are provided and the buyer inspects and accepts the items and approves payment. The common tasks associated with payment and contract close out are not shown for presentation clarity. The focus of our analysis is the interaction of the buyer, contract specialist and vendor, and we are analyzing only Government tasks.

With this Use Case representation of the Acquisition of Commercial Items specialization, the reader can see how the micro-technology view of software agents is integrated with the macro-process view of the DII. The decision node concept proves to be central to this integration. We now have a single process representation suitable for complexity analysis, which we now employ to identify candidate process tasks for software agents to perform.

D. PROCESS COMPLEXITY ANALYSIS

1. Defining Process-Complexity

Drawing from Kim [Ref. 18], we define process complexity in terms of requisite interaction between an effective agent system and its task environment. Any effective agent system (human or machine) must execute its assigned tasks within some task environment. In determining the complexity of the task, we consider both the nature of the task and the nature of the environment.

> A system must adapt its structure and behavior to thrive within a changing environment. The requisite complexity of a viable system is therefore determined by the complexity of the environment. ...It is clear that the structure and behavior of an intelligent system is defined not only by its goals but also by its surroundings. The more complex the environment, the more elaborate must be the requisite system interface and internal structure. [Ref. 18]

As noted above, an effective system [agent] must have the capacity to adapt in such a way that is suitable for the inherent complexity of the environment. If the environment and the nature of the task are both simple and routine, then the agent does not need to adapt. On the other hand, if the environment is complex then the agent must adapt to uncertainties, ambiguities and change.

Each decision node within the Use Case is specified through measurement based on four process-complexity dimensions: 1) Dynamism, 2) Temporal Association, 3) Accessibility and 4) Determinism. These dimensions are adapted from a popular text on Artificial Intelligence [Ref. 30: page 46]. Each dimension provides an indication of the

inherent complexity of the task, represented by the decision node, and the requisite behavioral sophistication of the agent (human or machine) responsible for the task.

Each process-complexity dimension is described in terms of its boundary values and through application. An assembly line process is used to provide clarification and promote understanding on how each dimension is applied. In our example the task goal is to execute the requisite decision processes to assemble a functional unit in a hypothetical factory.

Dynamism [Static - Dynamic] defines the temporal relationship between the decision node and the environment. Changes in the environment are relevant only if they affect the decision cycle duration or the decision outcome. An assembly line task that requires a worker to simply marry components out of a bin is a static process. On the other hand, if the conveyor belt is used to feed workers partially assembled components, then the decision cycle is compressed by the need to consider the movement of the belt and retrieve components as they appear (i.e., a dynamic task).

Temporal Association [Discrete - Sequential] defines the temporal relationship between decision-action pairs within each task instance. Discrete decisions are singular events and do not require planning. Decisions that require planning are sequential. In assembling a unit with multiple parts, where the order in which parts are married does not matter, then each action is independent of the others (i.e., a discrete task). When order is critical for proper operation then each action must be executed based on an ordered plan (i.e., a sequential task).

Accessibility [Accessible - Inaccessible] defines the extent in which sensory (interface) data can provide information sufficient for task execution. It indicates the nature of the boundary between the decision node and the environment in terms of perception. If sensory data is insufficient for decision execution, supplemental knowledge about the environment must be embedded within the decision node to ensure correct action is taken. If a worker in our example can determine the correctness of each part simply by inspection, then the task is accessible. If, on the other hand, a part must be separately tested and diagnosed in order to ensure correctness, then the decision node is inaccessible. In the inaccessible case, sensors alone are insufficient for task execution and reasoning using domain specific knowledge (e.g., a separate test and diagnosis task) is required to ensure correct execution.

Determinism [Deterministic - Non-Deterministic] defines the relationship between the decision node and the uncertainty of the action outcome. If probabilistic reasoning is

required to consider multiple probable outcomes prior to committing to a course of action, then the decision node is non-deterministic. If the outcome is certain based on the action taken, then the task is deterministic. A task that requires the assembly of multiple parts, having more then one possible way of being placed together, introduces some probability of an incorrect event occurring (i.e., a non-deterministic task). Therefore, the assembly worker must be aware of the possible outcomes and account for this uncertainty in executing the task. An idiot-proof assembly task, where parts have only one way of being placed together, is deterministic.

Using these four dimensions, we continue with our assembly line example and define a process-complexity space that encompasses the task spectrum from routine to complex. Both routine and complex tasks can be characterized using the four process-complexity dimensions. Because of the difficulty inherent in visualizing a four-dimensional space, we begin with only two dimensions - Accessibility and Determinism - and then build our four dimensional space from there.

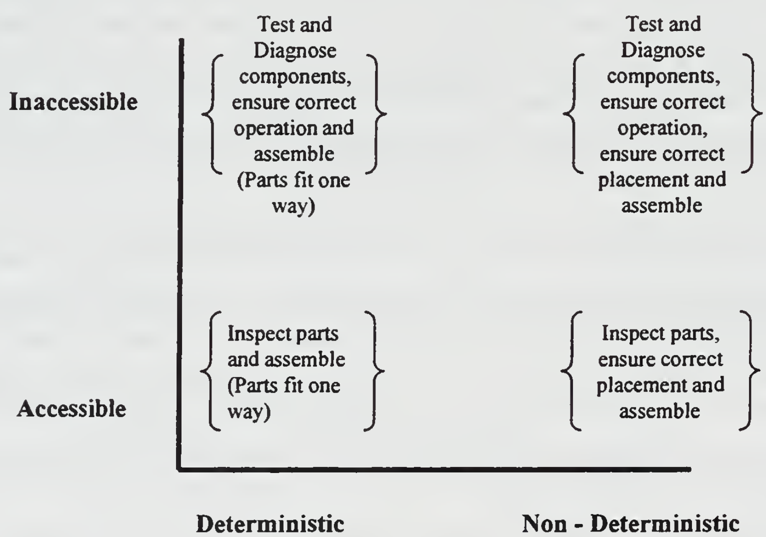


Figure 9: Accessibility verses Determinism Process-Complexity Space

The application of the Accessibility and the Determinism dimensions defines the relationships between the task and 1) sensory inputs and 2) type of reasoning needed to account for uncertainties in the world end-state, respectively. As we continue with our example in the two-dimensional space illustrated in Figure 9, we find the Accessible (bottom) row requires the agent to simply inspect the part for compliance. In the Inaccessible (top) row an agent is required to account for perception inadequacies by separately testing and diagnosing the part for compliance. The Determinism dimension indicates whether the agent must use probabilistic reasoning in determining the course of action. In the Deterministic (left) column, "idiot-proof" assembly ensures parts fit together only one way, whereas in the Non-deterministic (right) column, parts can fit together multiple ways, the latter of which requires probabilistic reasoning. Points in each of the four corners (i.e., boundaries) of this two-dimensional space are annotated in Figure 9 with short descriptions from our assembly example.

		Deterministic		Non - Deterministic	
		Static	**Dynamic**	**Static**	**Dynamic**
Inaccessible	**Sequential**	Test and Diagnose components, ensure correct operation, Plan order and assemble (Parts fit one way)	Retrieve unit (conveyor), Test and Diagnose components, ensure correct operation, Plan order and assemble (Parts fit one way)	Test and Diagnose components, ensure correct operation, Plan order, ensure correct placement and assemble	**Maximum Complexity** Retrieve unit (conveyor), Test and Diagnose components, ensure correct operation, Plan order, ensure correct placement and assemble
	Discrete	Test and Diagnose components, ensure correct operation and assemble (Parts fit one way)	Retrieve unit (conveyor), Test and Diagnose components, ensure correct operation and assemble (Parts fit one way)	Test and Diagnose components, ensure correct operation, ensure correct placement and assemble	Retrieve unit (conveyor), Test and Diagnose components, ensure correct operation, ensure correct placement and assemble
Accessible	**Sequential**	Inspect parts, plan order and assemble (Parts fit one way)	Retrieve unit (conveyor), Inspect parts, plan order and assemble (Parts fit one way)	Inspect parts, Plan order, ensure correct placement and assemble	Retrieve unit (conveyor), Inspect parts, Plan order, ensure correct placement and assemble
	Discrete	**Minimum Complexity** Inspect parts and assemble (Parts fit one way)	Retrieve unit (conveyor), Inspect parts and assemble (Parts fit one way)	Inspect parts, ensure correct placement and assemble	Retrieve unit (conveyor), Inspect parts, ensure correct placement and assemble

Figure 10: Process-Complexity Space

Building on this example and technique, we expand the process-complexity space to four dimensions and introduce the temporal elements of the task space - Dynamism and Temporal Association. Recall the discrete task does not require planning and the sequential task involves planning. Recall also in a static environment, time is irrelevant in decision outcome or duration and in a dynamic environment, the agent must determine when to stop deliberating and execute the decision as is. We identify minimum complexity (e.g., Accessible, Discrete, Static, Deterministic) as the lower left-hand corner of the four-dimensional space illustrated in Figure 10, where the agent must inspect and assemble parts that only fit one way. Note this point is also minimum complexity in the

two-dimensional space above. Maximum complexity (e.g., Inaccessible, Sequential, Dynamic, Non-deterministic) is defined at the point where an agent must retrieve moving parts from a conveyor belt, test and diagnose the part for proper operation, plan and order assembly, ensure the correct placement of parts and assemble. This is clearly a more complex task environment. Examples of 14 other points in this space are included for reference and should be self-explanatory. Each of the intermediate points lies somewhere in between the minimum and maximum complexity points identified above.

2. Analysis

We apply these heuristic measures of process complexity to the Use Case diagram representing the SAP Acquisition of Commercial Items specialization. As noted above, in our analysis, we assume a paperless procurement process and full DII enterprise support. The DII ensures secure reliable delivery of messages and handles all exceptions from link failures and other low-level errors. Based on the availability of the technical infrastructure, we focus on the process independent of the underlying infrastructure and assume the full benefits of automated systems support.

	Task	Process-Complexity	Task	Process-Complexity	
Requirements Phase	Generate Requirements	➢ Static ➢ Sequential ➢ Inaccessible ➢ Deterministic	Receive Vendor Response	➢ Static ➢ Discrete ➢ Accessible ➢ Deterministic	
	Create Purchase Request	➢ Static ➢ Discrete ➢ Accessible ➢ Deterministic	Evaluate Vendor's Offer	➢ Static ➢ Discrete ➢ Inaccessible ➢ Deterministic	
	Convey Purchase Request to Contract Specialist	➢ Static ➢ Discrete ➢ Accessible ➢ Deterministic	Award Contract	➢ Static ➢ Discrete ➢ Accessible ➢ Deterministic	
Pre-Award Phase	Validate Purchase Request	➢ Static ➢ Discrete ➢ Inaccessible ➢ Deterministic	Inspect & Accept Supplies and Services	➢ Static ➢ Discrete ➢ Inaccessible ➢ Non-Deterministic	Post-Award Phase
	Create Solicitation (Invitation to Bids)	➢ Static ➢ Discrete ➢ Accessible ➢ Deterministic	Make Payments (not shown)	➢ Static ➢ Discrete ➢ Accessible ➢ Deterministic	
	Solicit Vendors	➢ Static ➢ Discrete ➢ Accessible ➢ Deterministic	Close Out Award (not shown)	➢ Static ➢ Discrete ➢ Accessible ➢ Deterministic	

Table 3 : Summary of Analysis

A summary of our analysis is presented in Table 2 and the complete analysis is available in the Appendix. The table lists each process task required for the Acquisition of Commercial Items specialization and lists the corresponding complexity of each task in terms of the four process-complexity dimensions. For instance, the first process task - Generate Requirements - is assessed to have a complexity value (static, <u>sequential</u>, <u>inaccessible</u>, deterministic). Notice this value is above the minimum complexity defined above, due to its "sequential" and "inaccessible" ratings along the Temporal Association and the Accessibility dimensions, respectively. The Requirements Generation task requires acquisition planning and a translation of ill-defined user needs into a clearly defined service or product description. We assume a strict adherence to policies in executing the Requirements Generation task. This can be done through automated decision support aids. By doing so, the next task, Create Purchase Request denotes minimum complexity (e.g., static, discrete, accessible, deterministic) and is simply a

translation in data format by applying the results of the previous task. This is possible because all required information is available from the Generate Requirements planning task. Conveying the purchase request to the contract specialist is primarily a technology function and is accomplished via e-mail or some other messaging system. This completes the Requirements phase of the process.

In the Pre-Award phase the contract specialist validates the purchase request to ensure completeness and accuracy. As indicated before, automated systems can ensure form completeness, through programmed constraints ensuring each block within the form is addressed by the buyer. But an automated system cannot ensure accuracy. Accuracy is based on content and therefore the contract specialist cannot simply inspect the completed form, but must use specialized knowledge and reason about the nature of the inputs. The contract specialist must reason about the acquisition plan and ensure sufficient time is given for the solicitation-award-delivery process, ensure requirements are sufficiently described and ensure that entries are acceptable. This reasoning and application of specialized knowledge defines the task as inaccessible. The resulting process-complexity value of (static, discrete, inaccessible, deterministic) is assigned.

The Create Solicitation task is trivial and requires nothing more than formatting standard forms, inserting data and providing standard contract clauses according to the FAR and local policy. The contract specialist and buyer specify contract clause requirements, beyond standard clauses, in the Generate Requirements and Validate Purchase Request tasks. Contracting agency mailing lists and the DII Central Contractor Registration (CCR) database are used to locate and convey solicitations to perspective

vendors. Offers are then received by the contract specialist and secured until evaluation. In each case the task is routine and supported through the DII infrastructure, it is therefore valued (static, episodic, accessible, deterministic). Offers are evaluated based on price, other factors and solicitation compliance. The evaluation of other factors and ensuring solicitation compliance requires a reasoning ability, therefore the task is inaccessible. The resulting process-complexity value is (static, discrete, inaccessible, deterministic). The result of the Evaluate Vendors' Offer task is the selection of the offer that represents the "best value" to the Government. The award signing and notification to losing vendors are routine tasks and are valued (static, discrete, accessible deterministic).

In the post-award phase we assume the contract is administered in-house and the vendor is in full contract compliance. Supplies and services are provided by the vendor and inspected by the buyer for compliance to contract specifications. The provision and inspection tasks are outside the virtual environment and are inaccessible to software agents. Depending on the items delivered and the expertise of the inspector, the item(s) is (are) determined to be acceptable or unacceptable. In the absence of perfect knowledge, the determination is based on the probability of the item being acceptable or unacceptable and is therefore non-deterministic. The overall task is (static, discrete, inaccessible, non-deterministic). The buyer certifies receipt of acceptable items and approves payment. A notification is sent to the contract specialist and the Defense Finance and Accounting Service (DFAS). The Vendor sends an invoice to DFAS and payment is made through Electronic Funds Transfer (EFT). These tasks are messaging tasks and are valued (static, discrete, accessible, deterministic).

The SAP Acquisition of Commercial Items specialization is primarily based on the execution of routine tasks and the application of clearly defined policies from the FAR, FAR supplements and local policies. Tasks not routine in nature are those requiring planning, specialized knowledge or probabilistic reasoning for completion.

3. Task Suitability

In defining task suitability we assume that the greater the complexity of the process environment, the greater the need for a human agent. This is based in part on the primitive current state of agent technologies (micro-view) and the dynamic complexity of organizational processes (macro-view). In other words, we assume a human agent is more suitable for a process environment that demands high behavioral complexity in meeting the goals and objectives of the process. Although agent technologies and knowledge based systems have been successfully applied to a number of higher complexity tasks and environments [Ref. 17] [Ref. 19] [Ref. 10: page 30], we feel that until greater technological maturation, the use of software agents as an enterprise solution should be limited. Therefore, we take a conservative stance and use routine tasks of minimum process-complexity (e.g., static, discrete, accessible, deterministic) for determining task suitability for machine execution.

Supporting our position, we find at the routine edge (e.g., minimum process-complexity) a condition where human agent productivity decreases based on the routine nature of the task. Routine tasks are clearly defined tasks that require no abstract reasoning [Ref. 38: page 15]. In the field of job design, we find that human agents are more productive when process tasks have more variety and when there are opportunities

for learning [Ref. 9]. Therefore, the use of software agents for routine tasks allows the reassignment of human agents to more satisfying, complex and valuable efforts. With this, eight process tasks appear suitable for machine agents: Create Purchase Request, Convey Purchase Request to Contract Specialist, Create Solicitation, Solicit Vendors, Receive Vendor Response, Award Contract, Make Payments, Close Out Award. These eight tasks are evaluated for agent-based redesign opportunities in the following section. Before turning to this redesign discussion, we briefly describe the four tasks excluded by our analysis from consideration for agent-based redesign.

The results of our analysis show four tasks - Generate Requirements, Validate Purchase Request, Evaluate Vendor's Offer and Inspect & Accept Supplies and Services - are beyond our minimum process-complexity threshold. Actions that can be taken to reduce process-complexity (i.e., make them more suitable for software agents) for these tasks are as follows:

1. **Task 1: Generate Requirements (static, sequential, inaccessible, deterministic)** The sequential and inaccessible attributes applied to this task are linked. In order to accomplish acquisition planning, The agent combines organizational, procurement, process, technical, general and common sense knowledge in order to complete the task. This level of reasoning and knowledge and the complexity of the knowledge domain interactions require multiple knowledge representations and sophisticated reasoning processes. Further decomposition may provide a number of sub-goals leading to suitability, but at this level of abstraction this task is clearly unsuitable for a minimum process-complexity agent.

2. **Task 4: Validate Purchase Request (static, discrete, inaccessible, deterministic)** Accessibility demands that requirements are firm, stable and in machine-readable media and format. The use of standard product descriptions, such as those found in the *GSA Index of Federal Specifications, Standards and Commercial Item Descriptions*, and standard form entries would allow the environment to become accessible. The specification of standard product

descriptions and standard form entries would allow token matching and verification of entries through agent technologies.

3. **Task 7: Evaluate Vendor's Offer (static, discrete, inaccessible, deterministic)** The evaluation of vendors' offers is conducted based on price and other factors. Other factors "represent the key areas of importance and emphasis to be considered in the source selection decision and support meaningful comparison and discrimination between and among competing proposals." [Ref. 3: part 15.304] A number of quality related factors are past performance, compliance with solicitation requirements, technical excellence, management capability, personnel qualifications, and prior experience. [Ref. 3: part 15.304] Because of the large number of possible variations in defining each factor, it is doubtful that standardization would suffice. Therefore, this task requires extensive natural language abilities unavailable in current agent technologies.

4. **Task 10: Inspect & Accept Supplies and Services (static, discrete, inaccessible, non-deterministic)** This task is not conducted in the virtual environment and is therefore not applicable.

E. REDESIGNED SAP ACQUISITION OF COMMERCIAL ITEMS PROCESS

The use of agent technologies within the SAP process provides opportunities to redefine the task space for automated task execution and the reallocation and redefinition of work between software agents and human agents. One example of an agent-based redesign of this process is delineated in the Use Case diagram of Figure 11. To reiterate from above, this redesigned process incorporates software agents to perform each of the eight minimum-complexity tasks identified in the proceeding analysis.

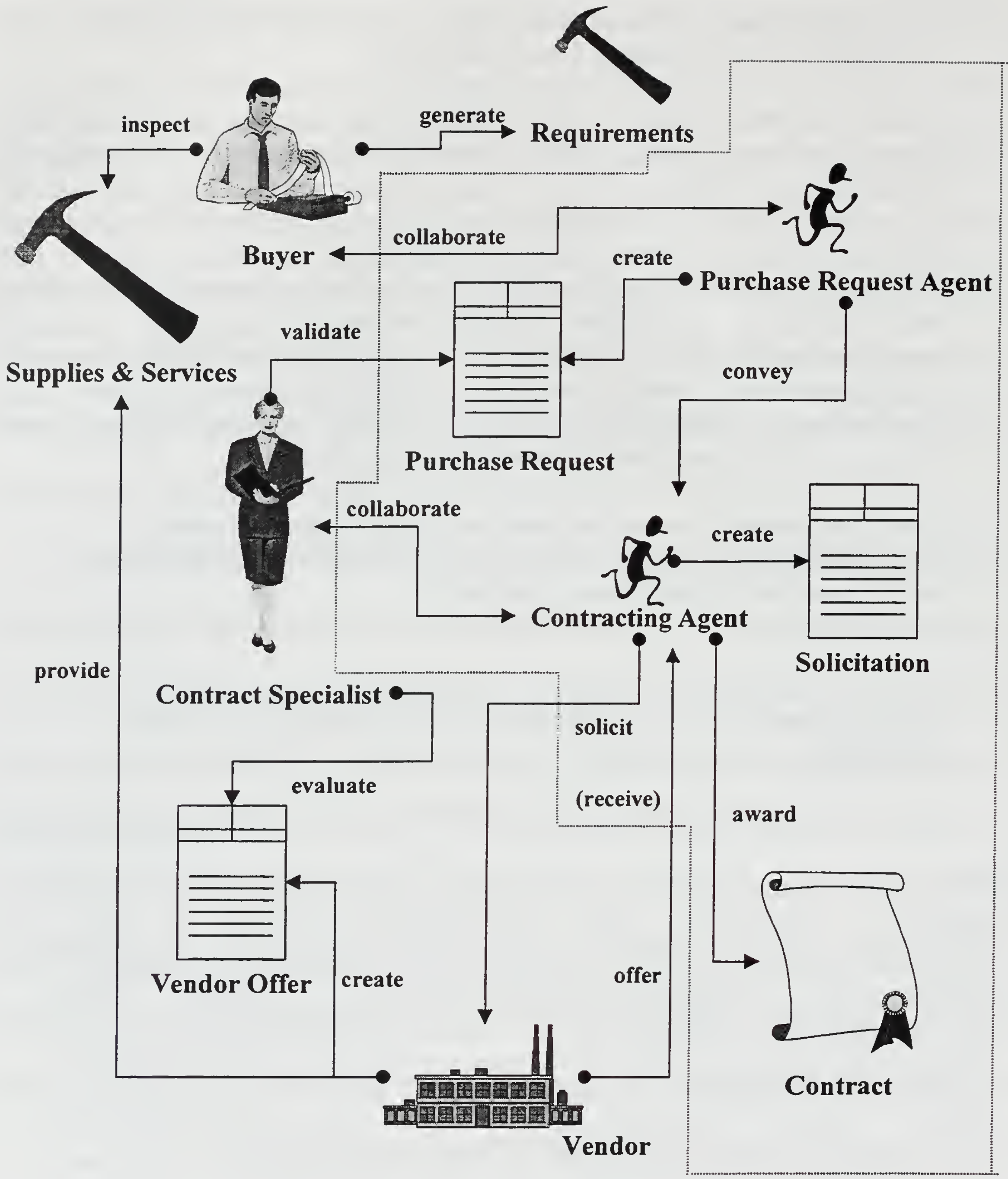


Figure 11: Redesigned SAP Commercial Item Acquisition Process

Using the software agent paradigm, the redesigned SAP Acquisition of Commercial Items process introduces a collaborative synergistic working relationship between

humans and machines. In our redesigned process, we depict a multi-agent system providing purchase request and solicitation creation and validation services, with solicitation distribution, receiving and secure storage capabilities. We envision the system also providing collaborative services to the buyer and the contract specialist through a web-based interface and the use of electronic mail (E-mail). For the buyer, the agent-based system has the role of advisor and assistant in creating purchase requests and the role of monitor in providing notification of procurement status. The collaborative relationship between the contract specialist and the agent-based system is more of a manger to subordinate relationship. The system would forward queries and exception cases to the contract specialist through E-mail, with the use of standard forms. In her new role, the contract specialist would act as manager and guide to direct the agent toward task completion.

Introducing these software agent capabilities into the process, we expect to see the following benefits:

- ❖ Task Automation (1) ⟶ Reduced Acquisition Lead Times
- ❖ Task Automation (2) ⟶ Increased Productivity
- ❖ Buyer Advice & Assistance ⟶ Reduced Errors and Learning Curve
- ❖ Buyer Notification ⟶ Increased Organizational Awareness
- ❖ Task Realignment ⟶ Greater Return on Human Assets

Even though we express the role and capabilities of agent-based systems according to the agent paradigm, at first glance it might appear that there is little difference between the use of agents and traditional software systems in terms of

embodying business logic and automating tasks. That is primarily based on our conservative stance in restricting task suitability. As noted before, software agents have been applied to a number of higher complexity tasks. But in each case a custom or ad-hoc approach was used and the scalability of the systems is questionable. Adhering to a minimum-complexity approach allows the use of existing software engineering methodologies and development technologies to realize the implementation and, based on the maturation of the technology, support a gradual incorporation of agents within the enterprise.

At this time, the real benefit of software agent systems is the acceptance of the agent paradigm. The software agent paradigm introduces the notion of automated systems as autonomous, intelligent and collaborative software entities, capable of interacting with the user, performing delegated tasks and cooperating with other agents and non-agent software systems to complete tasks. The acceptance of this paradigm requires continued research into the effects of such systems on the enterprise and its processes. As agent-based systems become more sophisticated, the level of autonomy, intelligence and collaboration will increase. This allows an increased delegation of responsibilities and interactivity in executing business processes. In our efforts, we show that based on limited process-complexity, software agents can be employed to redesign and innovate the SAP acquisition of commercial items within the enterprise. This is a small beginning. Only the future holds the key in defining the real value and richness of the software agent paradigm to DoD enterprise computing.

V. CONCLUSIONS AND RECOMMENDATIONS

A. INTRODUCTION

This research is based on taking a grounded look at agent technologies as they exist today. Our approach requires an iterative process, oscillating from the macro-process view of the DII enterprise to the micro-technology view of agent technologies, concluding with an integration of the views for analysis. First, we discuss the DII as the macro-view and the DoD vision of enterprise computing. It provides the context for agent incorporation and presents a vision of enterprise computing that is unprecedented in terms of access to multi-functional information resources, shared-data applications and a global communications network, providing ubiquitous connectivity. In addition, from the macro-view we better understand the implications of the DII's underlying technical strategy to support process innovation and the use of agent technologies. The micro-technology view is based on our conceptual definition of agency and the capabilities and limitations of current agent technologies. Finally, we integrate both views and use software agents as advanced information technology enablers to redesign DII Acquisition processes and determine the implications and benefits of software agents for process innovation and streamlining.

B. CONCLUSIONS

Our current strategy to transform and innovate the Defense Acquisition system must vigilantly seek out and apply advanced information technology products, such as software agents, in conjunction with sound process reengineering methodologies to bring

about increased organizational agility, flexibility, responsiveness and efficiency. We show that the integration of software agents within the DII acquisition system has the potential to radically change the dynamics and flow of the process domain by incorporating [intelligent] automated systems having process and task ownership responsibilities. The benefits of this include reduced acquisition lead times, increased productivity, reduced errors and learning curve, increased organizational awareness, and greater return on the application of human assets.

Software agents, as a computing abstraction, represent an evolution not a revolution in computing. Software agents are currently at a primitive stage in development but hold great promise for future advancements. However at this time, the technology offers minimal advantage over conventional systems except in a few narrowly defined application areas. In terms of general application, the best uses to date seem to be in the areas of information filtering, search automation functions and event monitoring. Even so, in spite of its immaturity, it represents a significant paradigm shift in how automated systems are viewed within the enterprise.

This paradigm shift influences the reengineering of business processes and the future design of process enabling and owning information systems. The agent metaphor indicates a desired level of human-computer interaction and functionality, allowing the delegation of tasks and an economical transfer of user beliefs, goals and desires, with a reasonable expectation of rational results. It is a higher level abstraction than is currently used in describing, specifying and implicitly understanding the behavior of complex

automated systems. In that it takes an intentional stance in defining system behavior using anthropomorphic terms such as belief, desire, intelligent and trustworthy.

In terms of enterprise computing, the agent metaphor indicates a shift from viewing automated systems as simply tools and process enablers to process owners and ultimately to intelligent autonomous virtual-workers. Processes suitable for software agents, in terms of process-complexity, can free human agents from routine tasks, enhance human capabilities and allow the organization to realign human resources to more valuable and critical functions.

C. RECOMMENDATIONS

1. Application Specific Agents

Until further maturation of agent technologies, the use of application specific agents, focused on providing user advice and assistance, is an area that has great research potential. Research that determines the level and type of interactivity that users desire and are willing to allow provides a foundation to creating truly interactive and intelligent systems. An extension to this area of research is to determine the best approach in linking application specific agents into organizational resources. Application specific agents could then seek out available knowledge sources within the organization and present it in the context of the application environment. Process related knowledge related to the services provided by the application would link the application to the broader organizational context and enhance the overall effectiveness of workers by linking activities within the application domain with organizational processes and constraints.

2. Structured Documents

As an enabling technology to agent systems, structured document technologies such as the Extensible Markup Language[4] (XML) should be investigated as a means of enriching the data search and processing capabilities of agent technologies. With standard-based structured documents processes that are currently inaccessible to agent technologies could become accessible for machine execution based on the use of open standards in creating business documents.

3. Knowledge Management

As a foundation to the deployment of intelligent systems a formal system of knowledge management within DoD should be investigated. A knowledge management strategy allows the free flow of lessons learned, studies, research and policies across the enterprise. The strategy should include provisions for the future translation of knowledge stores into machine executable formats by leveraging the work already done in creating a DoD data dictionary. Defining terms, objects, relationships and processes provides a knowledge-level infrastructure for creating a global environment for intelligent human and machine collaborative processing.

D. FUTURE RESEARCH

The following recommendations for future research are presented:

---

[4] XML - A World Wide Web Consortium (W3C) technology specification for structured documents developed to replacement Hypertext Markup Language (HTML).

1. Develop a standard model of coordination (i.e., communication, cooperation, collaboration, and negotiation) that can be used in creating multi-agent systems.
2. Create a small machine executable knowledge-base for executing procurement tasks.
3. Create a WWW software agent application capable of searching for and presenting XML documents, using standard languages and develop technologies.
4. Survey the acceptance and utility of current application suite technologies such as wizards, automatic spell checkers, and animated assistants to determine user acceptance and expectations.

APPENDIX A - PROCESS ANALYSIS

Use Case Name: **Award Contract for Commercial Items using Simplified Acquisition Procedures (Partial)**

Actor (Agent): **Buyer/Contract Specialist/Vendor**

1.1 DESCRIPTION:

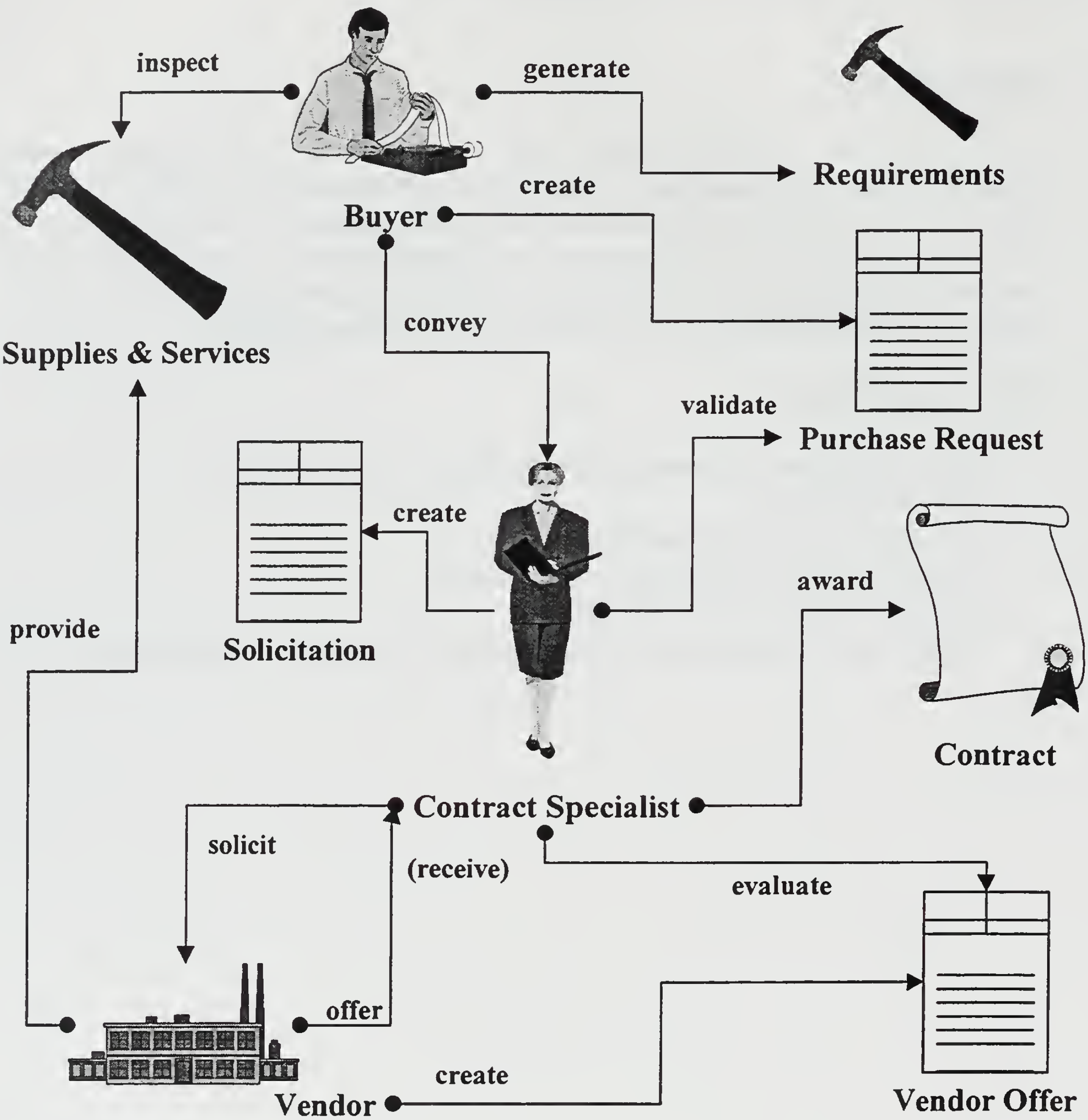
This Use Case describes the tasks executed by the Buyer/Contract Specialist/Vendor in awarding a contract for commercial items using the SAP process. We are only interested in the activities conducted by the government in completing the process. It is assumed that a paperless acquisition process is the standard and that the DII infrastructure is in place to support access to information resources, the presentation of data and the interconnection of agents.

1.2 PRE-CONDITIONS:

- A micro-purchase is unacceptable because of acquisition price. [FAR-Part 2.101]
- The item is a common commercial item.
- A pre-solicitation conference is not required.
- Contract Administration is in-house.
- Digital signatures are used for encryption and validation of authenticity.

1.3 DIAGRAM:

Use Case Name: *Simplified Acquisition Procedures - Commercial Purchase (Partial)*



1.4 DEFINITIONS OF SAP DOMAIN OBJECTS:

Definition of objects typically represented within the SAP process domain. All items are not represented in the (partial) Use Case diagram above.

1.4.1 **Buyer:** The Buyer (Agent) acts as the process owner and executes the process within the Buying Activity. The Buyer (Agent) must possess knowledge and skills equal to or greater then are necessary to operate within this process environment.

1.4.2 **Contract Specialist:** The Contract Specialist (Agent) acts as the process owner and executes the process within the Contracting Activity. The Contract Specialist (Agent) must possess knowledge and skills equal to or greater then are necessary to operate within this process environment.

1.4.3 **Vendor:** The Vendor acts as a source of products and product information for the Buyer. The Vendor acts as a recipient of Solicitations and a provider of responses to the Contract Specialist.

1.4.4 **Policies-Knowledge Base:** The Policies Knowledge Base is a store for enterprise, organization and process knowledge. This knowledge Base represents the stored knowledge of the enterprise in various forms and locations. The Acquisition Deskbook and the Acquisition Reform WWW site are examples of automated Knowledge Base resources.

1.4.5 **SHADE Information Directory:** The Defense Information Infrastructure information directory for shared data resources. This entity provides a standard application-programming interface (API) to programs operating within the DII.

1.4.6 **Contractor Registration Database:** The Acquisition Domain database containing the registered contractors providing products and services through EC/EDI mechanisms. This entity provides a database management system with a Structured Query Language (SQL) interface.

1.4.7 **Search Engine:** One of many WWW search engines providing URL specified resources based on keyword searches.

1.4.8 **Contractor On-Line Catalogs:** Vendor supported on-line catalogs providing access to Vendor specific information on products and services.

1.4.9 **Purchase Request:** A base form with necessary attachments provides the Contract Specialist information necessary to create a valid solicitation.

1.4.10 Technical Standards: An On-line resource providing information concerning technical standards.

1.4.11 **Contract:** A mutually binding legal relationship obligating the seller to furnish the supplies or services (including construction) and the buyer to pay for them. It includes all types of commitments that obligate the Government to an expenditure of appropriated funds and that, except as otherwise authorized, are in writing. [FAR-Part 2.101]

1.4.12 **Solicitation:** An announcement of the Government's intent to enter into a contract with responsible vendors.

1.4.13 Offer: A response to a solicitation that, if accepted, would bind the offeror to perform the resultant contract. [FAR-Part 2.101]

1.4.14 **Requirements:** A description of a buying agency's needs for products or services.

1.5 ANALYSIS CRITERIA - PROCESS-COMPLEXITY DIMENSIONS

- **Dynamism [Static - Dynamic]** defines the temporal relationship between the decision node and the environment. Changes in the environment are relevant only if it affects the decision cycle duration or the decision outcome.

- **Temporal Association [Discrete - Sequential]** defines the temporal relationship between decision-action pairs within each task instance. Discrete decisions are singular events and do not require planning. Decisions that require planning are sequential. Decisions are made considering a sequence of actions, in order to meet the goals and objectives of the task.

- **Accessibility [Accessible - Inaccessible]** defines the extent in which sensory (interface) data can provide information sufficient for task execution. It indicates the nature of the boundary between the decision node and the environment in terms of perception. If sensory data is insufficient for decision execution, supplemental knowledge about the environment must be embedded within the decision node to ensure correct action is taken.

- **Determinism [Deterministic - Non-Deterministic]** defines the relationship between the decision node and the uncertainty of the action outcome. If probabilistic reasoning is used to consider multiple probable outcomes prior to committing to a course of action, then the decision node is non-deterministic. If the outcome is certain based on the action taken then the task is deterministic.

1.6 DECISION NODE ANALYSIS:

1.6.1 TASK: Generate Requirements

Description:

This task is initiated with a validated need by the buying agency. The Buyer must identify potential sources of supply by searching the available On-line Acquisition Information Resources (e.g. Contractor Registration Database, Source list, Debarred lists, On-line Catalogs, other WWW resources). The Buyer must search available sources of information concerning commercial product availability and technical standards. The Buyer requests information from Vendors. The Buyer requests price, product data, delivery constraints, warranties, discounts and other pertinent data. The Vendor provides requested information to the Buyer. The Buyer executes a data analysis, synthesis and fusion process to determine what commercial products clearly fulfill the validated need [FAR-Part 11.002]. The output of this task is a plan that covers the following topics: Sources of supply, Source-selection factors, Contracting considerations (i.e. special clauses), Funding, Product or service descriptions, Logistics considerations (i.e. warranty, maintenance), Contract administration (i.e. inspection and acceptance criteria), Milestones for the acquisition cycle. [FAR-Part 7.105b]

Analysis:

Dynamism - The dynamism of the environment is driven by the urgency of the need. This type of procurement usually establishes a bid time, the time between solicitation and award, of 30 days. This time period is orders of magnitude beyond any concerns we might have about executing this process. Therefore in relative terms, we can consider this task as being STATIC.

Temporal Association - This is a planning task. SEQUENTIAL

Accessibility - This is required to act on a narrative description of the need and produce a valid requirements document. The narrative description does not provide sufficient information for the execution of this task. It is INACCESSIBLE.

Determinism - Since the need is for a commonly available commercial item, it is assumed market research and information processes within the

task will ensure a valid Purchase Request document for this purchase. The task is therefore DETERMINISTIC

1.6.2 TASK: Create Purchase Request

Description:

The Buyer creates the Purchase Request package and ensures the application of all relevant policies. All relevant information should be available from the previous task. The Buyer enters all relevant information and a validated funds citation. The Buyer attaches associated Vendor information and signs the Purchase Request. This process task is basically a fill in the blanks, routing, verification and approval task.

Analysis:

Dynamism - The dynamism of the environment is driven by the urgency of the need. This type of procurement usually establishes a bid time, the time between solicitation and award, at 30 days. This time period is orders of magnitude beyond any concerns we might have about executing this process. Therefore, we can consider this task as being STATIC.

Temporal Association - This task does not require planning and is therefore DISCRETE.

Accessibility - Information is presented in specific categories (i.e. Delivery date, Delivery address, contact person, warranty, product description, name brand equivalent, funds citation ... etc.). This task does not have to reason about the nature of the information in each category. It only has to place the information in the proper format. ACCESSIBLE

Determinism - The output of this task is a completed purchase request. Discounting human error, we assume that the Buying organization and the Contracting organization share the same policies. Therefore, the format is pre-determined and the task is DETERMINISTIC.

1.6.3 TASK: Convey Purchase Request to Contract Specialist

Description:

The Purchase Request package is conveyed to the Contract Specialist. The Contract Specialist performs necessary policy actions on the Purchase Request package such as date time stamping and logging.

Analysis:

Dynamism - The dynamism of the environment is driven by the urgency of the need. STATIC

Temporal Association - No planning is required. DISCRETE

Accessibility - The DII technical environment is standards based with standard application programming interfaces (API). A means of identifying the receiving party is all that is required to complete this task. The electronic transmission of data and the capability to time stamp and log an entry is ACCESSIBLE.

Determinism - From a process view, each task iteration will complete even if a negative response if returned. This task does not require probabilistic reasoning and traditional technology related exceptions can be handled in the programming interface. DETERMINISTIC

1.6.4 TASK: Validate Purchase Request

Description:

The Contract Specialist validates the Purchase Request by referring to relevant policies. The Contract Specialist ensures that the Purchase Request package is complete and accurate. This task requires the Contract specialist to ensure that the description of the requirement is complete, the planning is reasonable and that the funds citation is valid for this purchase. The Contract Specialist must exercise technical knowledge and common knowledge in executing this task.

Analysis:

Dynamism - The dynamism of the environment is driven by the urgency of the need. STATIC

Temporal Association - No planning is required. DISCRETE

Accessibility - The purchase request input is a standard form with attachments, but the Contract Specialist must ensure that the requirement is sufficiently described. An understanding of natural language and common sense knowledge is required for this task. INACCESSIBLE

Determinism - This task does not require probabilistic reasoning. DETERMINISTIC

1.6.5 TASK: Create Solicitation

Description:

The Contract Specialist refers to Policies and determines the correct procedures and content of the Solicitation [FAR-Part 12,13,14]. The solicitation is created, based on a standard format, standard contract clauses, additional clauses deemed necessary and adding specific information concerning the offer and evaluation process.

Analysis:

Dynamism - The dynamism of the environment is driven by the urgency of the need. STATIC

Temporal Association - This task is independent and is not temporally linked to the subsequent tasks. This task does not require planning. DISCRETE

Accessibility - The invitation for bid solicitation is based on standard formats [FAR-Part 14.2]. Based on a valid purchase request, the completion of the form is based on the entries placed on the purchase order and standard clauses and statements. ASSESSABLE

Determinism - This task does not require probabilistic reasoning. DETERMINISTIC

1.6.6 TASK: Solicit Vendor(s)

Description:

The solicitation is conveyed to the Vendor for action. This process is enabled by the Electronic Commerce (EC) /Electronic Data Interchange (EDI) infrastructure. The EC/EDI infrastructure allows access to the solicitation by all registered vendors. The solicitation is “pushed” to Vendors on specific Contract agency mailing lists.

Analysis:

Dynamism - The dynamism of the environment is driven by the urgency of the need. STATIC

Temporal Association - This task does not require planning. DISCRETE

Accessibility - The environment provides adequate exposure to the task for execution. No reasoning about the nature of the environment is required. ASSESSABLE

Determinism - This task does not require probabilistic reasoning and traditional technology related exceptions can be handled in the programming interface. DETERMINISTIC

1.6.7 TASK: Receive Vendor's Offer

Description:

The Purchase Request package is conveyed to the Contract Specialist. The Contract Specialist performs necessary policy actions on the Purchase Request package such as date time stamping and logging.

Analysis:

Dynamism - The dynamism of the environment is driven by the urgency of the need. STATIC

Temporal Association - No planning is required. DISCRETE

Accessibility - The DII technical environment is standards based with standard application programming interfaces (API). A means of identifying, the receiving party is all that required completing this task. The electronic transmission of data and the capability to time stamp and log an entry is ACCESSIBLE.

Determinism - From a process view, each task iteration will complete even if a negative response if returned. This task does not require probabilistic reasoning and traditional technology related exceptions can be handled in the programming interface. DETERMINISTIC

1.6.8 TASK: Evaluate Vendor(s) Offers

Description:

The Vendor prepares a response to the solicitation and conveys it to the Contract Specialist. The Contract Specialist validates the receipt of the Vendor's response. The offers are secured, opened and evaluated based on price and possibly other factors.

Analysis:

Dynamism - The dynamism of the environment is driven by the urgency of the need. STATIC

Temporal Association - This task does not require planning. DISCRETE

Accessibility - Solicitations requiring an evaluation based on other factors requires a reasoning ability. INACCESSIBLE

Determinism - This task does not require probabilistic reasoning. DETERMINISTIC

1.6.9 TASK: Award Contract

Description:

The best offer is selected for award and the contract is awarded. The losing Vendors are notified.

Analysis:

Dynamism - The dynamism of the environment is driven by the urgency of the need. STATIC

Temporal Association - This task does not require planning. DISCRETE

Accessibility - The environment provides adequate exposure to the task for execution. No reasoning about the nature of the environment is required. ACCESSIBLE

Determinism - This task does not require probabilistic reasoning. DETERMINISTIC

1.6.10 TASK: Inspect Supplies and Services

Description:

Supplies and services are provided and are inspected for compliance by the Buying agency. The Buying agency validates receipt of contractual items and authorizes payment.

Analysis:

Dynamism - The dynamism of the environment is driven by the urgency of the need. STATIC

Temporal Association - This task does not require planning. DISCRETE

Accessibility - The inspector must reason about the item delivered in relation to the requirements of the contract. It is unlikely a determination can be made simply on observation. INACCESSIBLE

Determinism - This task requires probabilistic reasoning. NON-DETERMINISTIC

1.6.11 TASK: Make Payments

Description:

The Contract Specialist forwards the payment authorization to DFAS. The vendor submits an invoice and payment is made to the Vendor.

Analysis:

Dynamism - The dynamism of the environment is driven by the urgency of the need. STATIC

Temporal Association - This task does not require planning. NON-DISCRETE

Accessibility - The DII technical environment is standards based with standard application programming interfaces (API). A means of identifying, the receiving party is all that required completing this task. The electronic transmission of data and the capability to time stamp and log an entry is ACCESSIBLE.

Determinism - This task does not require probabilistic reasoning and traditional technology related exceptions can be handled in the programming interface. DETERMINISTIC

1.6.12 TASK: Close Out AWARD

Description:

Supplies and services are provided and are inspected for compliance by the Buying agency. The Buying agency validates receipt of contractual items and authorizes payment. Payment is made to the Vendor. The contract is closed out.

Analysis:

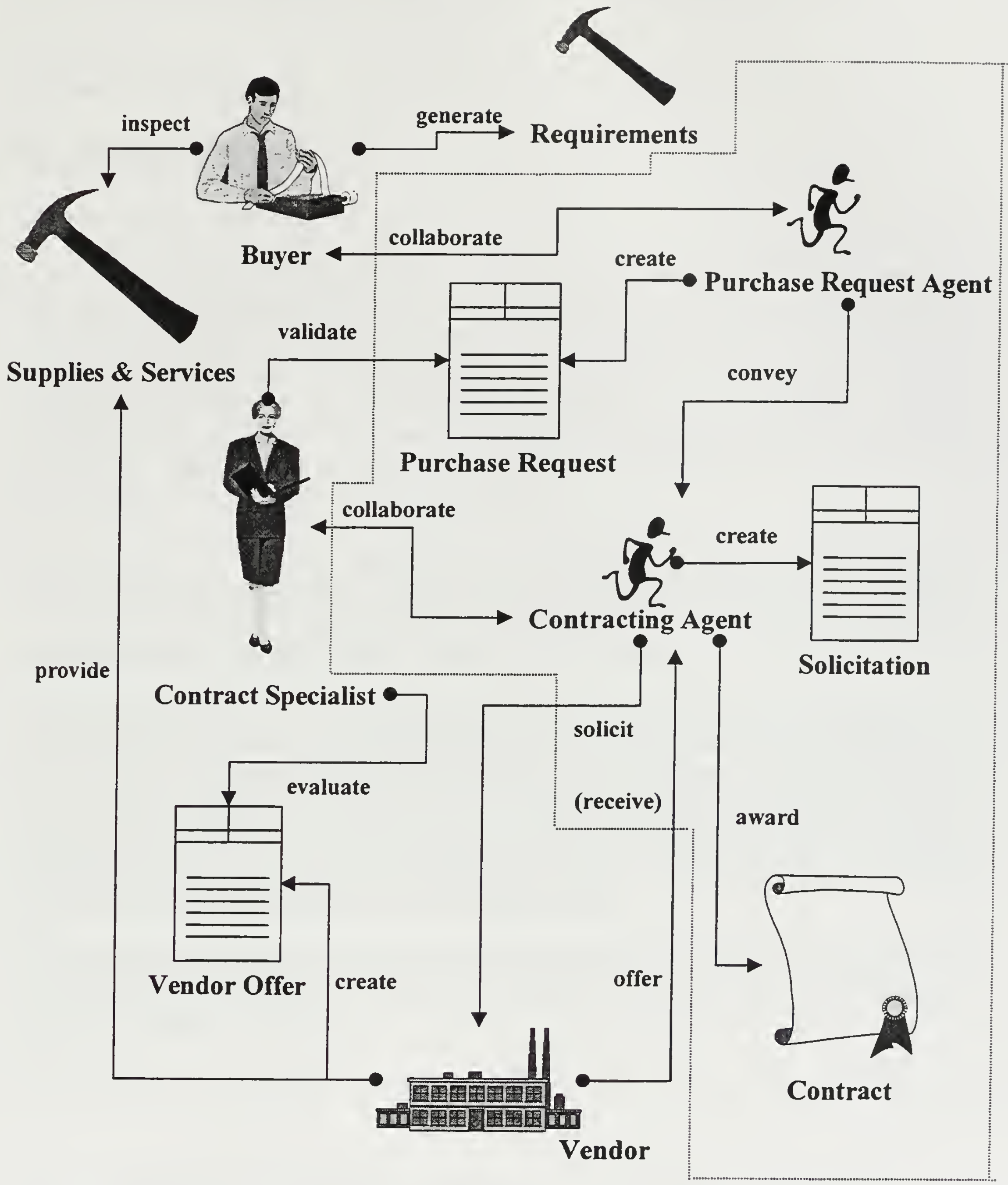
Dynamism - The dynamism of the environment is driven by the urgency of the need. STATIC

Temporal Association - This task does not require planning. DISCRETE

Accessibility - The DII technical environment is standards based with standard application programming interfaces (API). A means of identifying, the receiving party is all that required completing this task. The electronic transmission of data and the capability to time stamp and log an entry is ACCESSIBLE.

Determinism - This task does not require probabilistic reasoning and traditional technology related exceptions can be handled in the programming interface. DETERMINISTIC

1.7 REDESIGNED PROCESS



APPENDIX B - AGENT DEVELOPMENT TOOLS

Commercial Products

AgentBuilder®

Reticular Systems, Inc.

AgentBuilder® is an integrated tool suite for constructing intelligent software agents. AgentBuilder consists of two major components - the Toolkit and the Run-Time System. The AgentBuilder Toolkit includes tools for managing the agent-based software development process, analyzing the domain of agent operations, designing and developing networks of communicating agents, defining behaviors of individual agents, and debugging and testing agent software. The Run-Time System includes an agent engine that provides an environment for execution of agent software.

Agents constructed using AgentBuilder communicate using the Knowledge Query and Manipulation Language (KQML) and support the performatives defined for KQML. In addition, AgentBuilder allows the developer to define new inter-agent communications commands that suit his particular needs.

All components of both the AgentBuilder Toolkit and the Run-Time System are implemented in Java. This means that agent development can be accomplished on any machine or operating system that supports Java and has a Java development environment. Likewise, the agents created with the AgentBuilder Toolkit are Java programs so they can be executed on any Java virtual machine. Software developers can create powerful intelligent agents in Java that execute on a wide variety of computer platforms and operating systems.

The AgentBuilder toolkit is designed to provide the agent software developer with an integrated environment for quickly and easily constructing intelligent agents and agent-based software.

AgenTalk: Describing Multiagent Coordination Protocols

NTT and Ishida

AgenTalk is a coordination protocol description language for multi-agent systems.

In the distributed artificial intelligence area, many coordination protocols such as the contract net protocol have been proposed, and many application-specific protocols will be required as more software agents start to be built. AgenTalk allows coordination protocols to be defined incrementally and to be easily customized to suit application domains by incorporating an inheritance mechanism.

AgenTalk is being co-developed by NTT Communication Science Laboratories and Ishida Laboratory, Department of Information Science, Kyoto University. The software is written in Common Lisp and is only distributed in Japan.

ABE

IBM

IBM's Agent Building Environment (ABE) is a toolkit for software developers that eases building applications based on intelligent agents and simplifies adding agents to an existing applications. In the current version, the intelligent agent watches for a certain condition, decides what to do based on a set of rules, and triggers an action as a result.

This developer kit provides a number of pre-built parts which make it easy to add agent technology to applications. The architecture for the agent is based on reasoning engine and adapter technologies from IBM's T.J. Watson Research Lab. "Adapters" or interfaces allow the agent to interact with the rest of the world. For instance, the HTTP adapter provided with ABE interfaces with the world-wide-web. The NNTP adapter interfaces with internet USENET news services, and the timer adapter allows events to be triggered based on time. The developer can construct custom adapters; guidelines and a sample adapter are also provided. Custom adapters can be written in either C++ or Java .

Versions are available for OS/2, Windows, AIX and OS/390.

Aglets

IBM Japan

An aglet is a Java object that can move from one host on the Internet to another. That is, an aglet that executes on one host can suddenly halt execution, dispatch to

a remote host, and resume execution there. When the aglet moves, it takes along its program code as well as its state (data). A built-in security mechanism makes it safe for a computer to host untrusted aglets.

Concordia

Mitsubishi Electric
Information Technology Center America

CONCORDIA is a full-featured framework for development and management of network-efficient mobile agent applications for accessing information anytime, anywhere and on any device supporting Java. With Concordia, applications can:

- Process data at the data source.
- Process data even if the user is disconnected from the network.
- Access and deliver information across multiple networks (LANs, Intranets and Internet).
- Use wire-line or wireless communication.
- Support multiple client devices, such as Desktop Computers, PDAs, Notebook Computers, and Smart Phones.

Java Intelligent Agent Library

Bits & Pixels

The Intelligent Agent Library provides an intelligent agent framework that includes extensive facilities for agent communications and for building larger agent assemblies. There is a KQML-based agent framework and many examples illustrating agents that perform activities for web-enabled applications. The library also supports mobile agents.

Libraries are available from Bits & Pixels, 8601 Del Mesa, Austin, Texas 78759

KAFKA: Yet Another Multi-Agent Library for Java

Fujitsu

Kafka is yet another agent library designed for constructing multi-agent based distributed applications. Kafka is a flexible, extendable, and easy-to-use Java class library for programmers who are familiar with distributed programming. It is based on Java's RMI and has the following added features:

Runtime Reflection:

> Agents can modify their behavior (program codes) at runtime. The behavior of the agent is represented by an abstract class Action. It is useful for remote maintenance or installation services.

Remote Evaluation:

> Agents can receive and evaluate program codes (classes) with or without the serialized object. Remote evaluation is a fundamental function of a mobile agent and is thought to be a push model of service delivery.

Distributed Name Service:

> Agents have any number of logical names that don't contain the host name. These names can be managed by the distributed directories.

Customizable security policy

> A very flexible, customizable, 3-layered security model is implemented in Kafka.

100% Java and RMI Compatible:

> Kafka is written completely in Java. An agent is a Java RMI server object itself. Therefore, agents can directly communicate with other RMI objects.

LiveAgent - Web Automation Software

AgentSoft Ltd.

AgentSoft's Java-based LiveAgent Pro provides an easy to use, powerful, flexible, and extensible way to automate Web activity. LiveAgent Pro is used to create Internet/Intranet scripts using a recording environment (like a high-level macro

recorder or automated testing tool). A developer performs a sequence of Web operations in his/her browser, and those actions are automatically saved by LiveAgent Pro as a script (or "agent").The completed script can then be run by the user or scheduled for automatic launching.

Microsoft Agents

Microsoft Corporation

Microsoft Agent is a set of programmable software services that supports the presentation of interactive animated characters within the Microsoft Windows interface. Developers can use characters as interactive assistants to introduce, guide, entertain, or other wise enhance their Web pages or applications in addition to the conventional use of windows, menus, and controls.

Microsoft Agent enables software developers and Web authors to incorporate a new form of user interaction, known as conversational interfaces, that leverages natural aspects of human social communication. In addition to mouse and keyboard input, MicrosoftAgent includes optional support for speech recognition so applications can respond to voice commands. Characters can respond using synthesized speech, recorded audio, or text in a cartoon word balloon.

The conversational interface approach facilitated by the Microsoft Agent services does not replace conventional graphical user interface (GUI) design. Instead, character interaction can be easily blended with the conventional interface components such as windows, menus, and controls to extend and enhance your application's interface.

Microsoft Agent's programming interfaces make it easy to animate a character to respond to user input. Animated characters appear in their own window, providing maximum flexibility for where they can be displayed on the screen. Microsoft Agent includes an ActiveX control that makes its services accessible to programming languages that support ActiveX, including Web scripting languages such as Visual Basic Scripting Edition (VBScript). This means that character interaction can be programmed from HTML pages.

Microsoft Agent requires Microsoft Windows, Internet Explorer and ActiveX.

Odyssey (Son of Telescript)

General Magic

Odyssey is an agent system implemented as a set of Java class libraries that provide support for developing distributed, mobile applications. Odyssey provides Java classes for agents and places. Odyssey agents are Java threads. They are created by subclassing the Odyssey agent class or the Odyssey worker class.

Voyager

Object Space

Voyager is a 100% Java agent-enhanced Object Request Broker (ORB). It combines the power of mobile autonomous agents and remote method invocation with complete CORBA support and comes complete with distributed services such as directory, persistence, and publish subscribe multicast. Voyager allows Java programmers to quickly and easily create sophisticated network applications using both traditional and agent-enhanced distributed programming techniques.

Voyager uses regular Java message syntax to construct remote objects, send them messages, and move them between applications. Voyager allows agents (i.e, autonomous objects) to move themselves and continue executing as they move. In this way, agents can act independently on the behalf of a client, even if the client is disconnected or unavailable. This approach is particularly valuable in any type of workflow or resource automation.

Research Projects

The Agent Building Shell: Programming Cooperative Enterprise Agents

Enterprise Integration Laboratory
University of Toronto

This project is developing an Agent Building Shell that provides several reusable layers of languages and services for building agent systems: coordination and communication languages, description logic-based knowledge management,

cooperative information distribution, organization modeling and conflict management. The approach is being used to develop multi-agent applications in the area of manufacturing enterprise supply chain integration.

Agent TCL

Dartmouth University

Agent Tcl is a tool for developing transportable agent systems. The transportable agents are created using the Tool Command Language (Tcl). Tcl is an embeddable scripting language that is highly portable, highly popular and freely available.

The agents migrate from machine to machine using the jump command. Execution resumes on the destination machine at the statement immediately after the jump is completed. Modifications to the Tcl core allow the capture of the complete internal state of an executing script. Migrating agents are encrypted and authenticated using Pretty Good Privacy (PGP). Access restrictions are imposed on the agent based on its authenticated identity. Safe Tcl enforces the access restrictions.

In addition to migration, Agent Tcl supports message passing. Agents can clone themselves and the system provides rudimentary security features. Each agent on a particular machine has a unique integer ID and a unique symbolic name. Agents specify a recipient agent by specifying the recipient's machine and either the recipient's integer ID or the recipient's symbolic name.

The research project is addressing issues involving debugging, privacy, security, mobile agent management, networking resources and performance. Agent Tcl has two components: a modified Tcl interpreter that execute Tcl agents and a server which runs on every machine that can receive a transportable agent.

Cable - Intelligent Agents

Logica Corporation

Cable is a generic system architecture developed by Logica as part of the GRACE Consortium. Cable can be used to develop and execute distributed applications that are based on the metaphor of multiple, cooperating intelligent agents.

Cable provides the user with an Agent Definition Language (ADL), for defining agents, and a parser known as the Scribe, for compiling agent definitions written in ADL into agent applications. Agents are developed using ADL and C++. ADL allows developers to use Cable without worrying about underlying detail, providing a language with a level of abstraction close to that of the agents with which an application is designed. Inter-agent communication over a local area network is handled using ORBIX, an implementation of the CORBA 2.0 standard.

Cybele: An Infrastructure for Autonomous Agents

Intelligent Automation, Inc.
2 Research Place, Suite 202
Rockville, MD 20850

This infrastructure supports a number of services for agent-based applications that most platforms do not provide. These include: (i) Agent creation and deployment over a network of varied platforms, (ii) a message addressing scheme for agent communication which is independent of the location of a sending or receiving agent, (iii) the accumulation of messages intended for a currently busy recipient agent, (iv) the proper conversion of message data across platforms, (v) multicasting, broadcasting, and peer-to-peer messaging, and (vi) the migration of agents across processors for performance optimization and/or fault tolerance.

Cybele is easy to use and allows application developers to focus on developing agents, and not on implementing agent infrastructures. Cybele's strong points are high performance, scalability, and support for rapid development.

---

dMARS

Australian Artificial Intelligence Institute Ltd.

dMARS™ is an agent-oriented development and implementation environment for building complex, distributed, time-critical systems. Designed for rapid configuration and ease of integration, it facilitates system design, maintenance and re-engineering. This product is based on the older Procedural Reasoning System (PRS) developed by SRI International (California). dMARS takes advantage of the latest research into multi-agent, real-time reasoning.

dMARS is suited to the development of any application that requires both

proactive goal directed behavior and reliable time-critical response to change. It is particularly well suited to applications where a large number of complex but well-defined procedures or tactics exist for accomplishing particular tasks in a variety of situations.

dMARS was designed with the issues of robustness, efficiency and user-extensibility in mind. It provides a sophisticated suite of graphical tools for development and debugging. These tools not only provide an intuitive interface, but address the specific issues involved with large-scale development.

dMARS provides a high-level, graphical plan language for specifying complex, context-dependent processes and tactics. dMARS is written in C and C++ and will execute on a variety of UNIX platforms. dMARS is available from the Australian Artificial Intelligence Institute Ltd.,Level 6, 171 La Trobe Street, Melbourne 3000 Australia.

Gypsy

Technical University of Vienna

The Gypsy Project utilizes Java for the implementation of a flexible environment for experimenting with mobile agent programming. It is intended for application in Internet information retrieval, Internet commerce, mobile computing, and networks network management.

Infospiders

University of California San Diego - Computer Science Dept.

By Filippo Menczer and Rik Belew

InfoSpiders (aka ARACHNID: Adaptive Retrieval Agents Choosing Heuristic Neighborhoods for Information Discovery)

This project features an artificial life inspired model using endogenous fitness for information retrieval in large, dynamic, distributed, heterogeneous databases, such as the WWW. A population of agents is evolved under density dependent selection for the task of locating information for the user. The energy necessary for survival is obtained from both environment and user in exchange for relevant information. By competing for energy, the agents robustly adapt to their

environment and are allocated to efficiently exploit their shared resources.

JAFMAS

University of Cincinnati

JAFMAS provides a framework to guide the development of multi-agent systems along with a set of classes for agent deployment in Java. The framework is intended to help beginning and expert developers structure their ideas into concrete agent applications. It directs development from a speech-act perspective and supports multicast and directed communication, KQML or other speech-act performatives and analysis of multi-agent system coherency and consistency. The JAFMAS project provides a good comparison of agent tools with a particular emphasis on mobile agent projects.

JATLite

Stanford University

JATLite is a set of Java packages that make it easy to build multi-agent systems using Java. JATLite provides a basic infrastructure in which agents register with an Agent Message Router facilitator using a name and password, connect/disconnect from the Internet, send and receive messages, transfer files, and invoke other programs or actions on the various computers where they are running. JATLite facilitates construction of agents that send and receive messages using the emerging standard communications language, KQML (see http://www.cs.umbc.edu/kqml/ for the current KQML standard). The communications are built on open Internet standards, TCP/IP, SMTP, and FTP.

Kasbah

Agent-mediated Electronic Commerce (AmEC) Initiative Massachusetts Institute of Technology

Kasbah is an ongoing multi-agent research project to develop an agent-mediated

electronic commerce system. A user wanting to buy or sell a product or service will create an agent, give it some strategic direction, and send it off into the agent marketplace. Kasbah agents pro-actively seek out potential buyers or sellers and negotiate with them on their creator's behalf. Each agent's goal is to make the "best deal" possible, subject to a set of user-specified constraints, such as a desired price, a highest (or lowest) acceptable price, and a date to complete the transaction.

LALO

CRIM - Centre de recherche informatique de Montréal Canada

LALO is a programming environment, which permits the development of multi-agent systems. The architecture is extensible and allows creating multi-agent systems including reactive agents and deliberate agents. In addition, LALO permits the definition of agents using this new programming paradigm. The inter-agent communication language used is KQML ("Knowledge Query and Manipulation Language"). LALO is an Agent Oriented Programming (AOP) language and a framework for developing intelligent multi-agent systems. A program written in LALO is translated into C++ source code, and then compiled with a C++ compiler.

Mole

University of Stuttgart

This is a research project investigating mobile agents.

The basic concepts for Mobile Agent systems have been developed. Furthermore a prototypical implementation has been developed that shows the feasibility of this approach. This prototype adds mechanisms for migration and communication using the Java programming language.

Process Link

Stanford University

The objective of this project is to enable design engineers to track and coordinate their design decisions with each other even when not co-located or working with the same software.

Approach

The project is developing an agent-based framework consisting of generic agents and a message protocol for integrating multidisciplinary engineering software and managing distributed design projects. This framework allows them to "wrap" legacy software with backend code that will disturb the existing software interface as little as possible while providing useful coordination functions.

They are using a "weak" agent technology in which the wrapped software become agents in that they send messages corresponding to interaction semantics, but they don't necessarily have to be "smart" or conform to any particular theory of agent construction. The only commitment is to send messages conforming to a defined set of interactions.

Sodabot

MIT Artificial Intelligence Lab

Sodabot is a general-purpose software agent user-environment and construction system. Its primary component is the basic software agent- a computational framework for building agents which is essentially an agent operating system. This project developed a new language for programming the basic software agent whose primitives are designed around human-level descriptions of agent activity. Using this programming language, users can implement a wide-range of typical software agent applications, e.g. personal on-line assistants and meeting scheduling agents.

Swarm

Sante Fe Institute

Swarm is a software package for multi-agent simulation of complex systems. Swarm is intended as a tool for researchers in a variety of disciplines, especially artificial life. The basic architecture of Swarm is the simulation of collections of concurrently interacting agents: with this architecture, we can implement a large variety of agent based models. The initial target is Unix machines running GNU Objective C and X windows: the source code is freely available under GNU Licensing terms.

AgentBuilder Internet WWW Page: http://www.agentbuilder.com

LIST OF REFERENCES

1. Chavez A., Dreilinger D., Guttman R., and Maes P., *A Real-Life Experiment in Creating and Agent Marketplace*, Software Agents and Soft Computing, Pages 160-179, Lecture Notes in Artificial Intelligence, Berlin, Springer-Verlag, 1997

2. Davenport T., *Process Innovation: Reengineering Work through Information Technology*, Harvard Business School Press, 1993

3. Defense Acquisition Deskbook, *Federal Acquisition Regulation*, Version 2.2, December 1997

4. Defense Information Systems Agency, *Defense Information Infrastructure (DII) - Shared Data Environment - Capstone Document*, Version 1.0, July 1996

5. Defense Information Systems Agency, *Defense Information Infrastructure Master Plan*, Version 6.0, 1997

6. Defense Information Systems Agency, *Integration and Runtime Specification (I&RTS)*, Version 3.0, 1997

7. Erwin W.B., *Automated small-purchasing streamlines operations, upgrades the procurement function, and empowers program offices*, http://www-far.npr.gov/References/Best_Pract_Docs/paten.html

8. Gilbert D. and Janca P., *IBM Intelligent Agents*, IBM Corporation, http://www.raleigh.ibm.com/iag./iagwp1.html, 1996]

9. Hackman J. and Oldham G., *Work Redesign*, page 256, Addison-Wesley, 1980

10. Hayes-Roth F. and Jacobstein N., *The State of Knowledge-Based Systems*, Communications of the ACM Vol. 37 No. 3, 1994

11. Hayes-Roth F., *Artificial Intelligence - What Works and What Doesn't*, pages 99-113, AI Magazine, Summer 1997

12. http://www.agentbuilder.com/AgentTools

13. IEEE Internet Computing, *Pattie Maes on Software Agents*, Page 11, July-August 1997

14. Intelligent Agents Group, *Software Agents: A Review*, http://www.cs.tcd.ie/Brenda.Nangle/iag.html, 1997

15. Joint Chiefs of Staff, *Joint Vision 2010: America's Military Preparing for Tomorrow*, 1996

16. Joint Staff, *C4I For The Warrior: Committed, Focused and Needed*, 1992

17. Jones R., Wray R., Lent M., and Laird J., *Planning in the Tactical Air Domain*, AI Laboratory University of Michigan, {rjones, wrayre, vanlent, laird}@eecs.umich.edu, 1995

18. Kim, Steven H., *Designing Intelligence: A Framework for Smart Systems*, Oxford University Press, New York, 1990

19. Larsson J. and Hayes-Roth B., *Guardian: An Intelligent Autonomous Agent for Medical Monitoring and Diagnosis*, IEEE Intelligent Systems - IEEE Computer Society, Jan/Feb 1998

20. Laufmann S.C., *Towards Agent-based Software Engineering for Information-Dependent Enterprise Applications*, IEE Proceedings - Software Engineering, Vol. 144 No. 1, Feb. 1997

21. Logistics/Life Cycle Information Integration Office, Electronic Commerce Project FACT Sheet, 1997

22. Mitaim S. and Kosko B., *Neural Fuzzy Agents for Database Search, Cooperative Information Agents*, pages 159-170, Lecture Notes in Artificial Intelligence, Berlin, Springer-Verlag, 1997

23. Negroponte N., *Being Digital*, page 102, Alfred A. Knopf Inc., New York, 1995

24. Nwana H.S., Lee L. and Jennings N.R., *Co-ordination in Multi-Agent Systems*, Software Agents and Soft Computing, Page 42-58, Lecture Notes in Artificial Intelligence, Berlin, Springer-Verlag, 1997

25. O'Brien P.D. and Wiegand M.E., Lecture Notes in AI: Software Agent and Soft Computing, *Agents of Change in Business Process Management*, Springer-Verlag, Berlin, 1997

26. OUSD(A&T) Information Management Executive Briefing, *Global Business Interfaces: Integrating EC Business Applications in Global Defense*, 1997

27. Pan, Jeff J. & Tenenbaum, Jay, *An Intelligent Agent Framework for Enterprise Integration*, IEEE Transactions on Systems, Man and Cybernetics, Vol. 21, No. 6, Nov/Dec 1991

28. Puppe F. and Gappa U., *Towards Knowledge Acquisition by Experts, Industrial and Engineering Applications of Artificial intelligence and Expert Systems*, Pages 546-555, Lecture Notes in Artificial Intelligence, Berlin, Springer-Verlag, 1992

29. Rauch-Hindin W., *A Guide to Commercial Artificial Intelligence*, Prentice Hall Englewood Cliffs NJ, 1988

30. Russell S. and Norvig P., *Artificial Intelligence: A Modern Approach*, Prentice Hall Englewood Cliffs NJ, 1995

31. Shoham, Yoav, *Agent-oriented Programming*, Page 51-92, Artificial Intelligence Vol. 60, 1993

32. St. Moritz M.E., *The Application of Reengineering to the Acquisition Planning Process for a Major Weapon System: A Case for Information Technology*, Masters Thesis Naval Postgraduate School, 1997

33. Sycara K., Pannu A., Williamson M. and Zeng D., *Distributed Intelligent Agents*, IEEE Expert, 1996

34. The American Heritage Dictionary - Second College Edition, Houghton Mifflin Co., Bosten, 1976

35. Tzafestas E., *Agentifying the Process: Task-based or Robot-based Decomposition*, IEEE, 1994

36. University of Michigan, *A Survey of Cognitive and Agent Architectures*, http://ai.eecs.umich.edu/cogarch0/index.html

37. Watt SNK, *Artificial Societies and Psychological Agents*, Software Agents and Soft Computing, Page 27-39, Lecture Notes in Artificial Intelligence, Berlin, Springer-Verlag, 1997

38. Wooldridge M. and Jennings N., *Agent Theories, Architectures and Languages: A Survey, Intelligent Agents*, Lecture Notes in Artificial Intelligence, Berlin, Springer-Verlag, 1994

39. Wooldridge M. and Jennings N., *Intelligent Agent: Theory and Practice*, The Knowledge Engineering Review, No. 2, Pages 115-152, 1995

40. Wooldridge M., IEE Proceedings on Software Engineering, Vol. 144 , No. 1, *Agent-based Software Engineering*, Pages 26-37, 1997

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center...2
   8725 John J. Kingman Road, Suite 0944
   Fort Belvoir, VA 22060-6218

2. Dudley Knox Library...2
   Naval Postgraduate School
   411 Dyer Road
   Monterey, CA 93943-5101

3. Defense Logistics Studies Information Exchange.......................................1
   U.S. Army Logistics Management College
   Fort Lee, VA 23801-6043

4. Professor Mark E. Nissen..1
   Code SM/NI
   Naval Postgraduate School
   Monterey, CA 93943-5000

5. Tung Bui, Ph.D., Dr.rer.pol..1
   Matson Distinguished Professor of Global Business
   The University of Hawaii
   College of Business Administration
   2404 Maile Way, Room E303
   Honolulu, HI 96822

6. Major Jerome Hudson...1
   1313 Rose Ave.
   Killeen, TX 76541